NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DIVER RELATIVE UUV NAVIGATION FOR JOINT HUMAN-ROBOT OPERATIONS

by

Andrew T. Streenan

September 2013

Thesis Advisor:	Noel Du Toit
Second Reader:	Doug Horner

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2013	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE DIVER RELATIVE UUV NAVIGATION FOR JOINT HUMAN-ROBOT OPERATIONS			**5. FUNDING NUMBERS**
6. AUTHOR(S) Andrew T. Streenan			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

A novel application for Autonomous Underwater Vehicles (AUVs) is considered here: a robotic diver assistant that enables close-quarters robotic operations with human divers. A robotic diver assistant has the potential to improve the efficiency, effectiveness and safety of diver operations. The robot diver assistant must share the operating environment with human divers, navigate relative to the environment to reach a specified site location (along with moving divers), and then maneuver among the mostly static divers as they perform their tasks on location. The robot operates in three unique scenarios: station-keeping, diver-following (shadowing), and diver-leading (vectoring). Various strategies for navigating among divers while ensuring diver safety are investigated. A reactive strategy, based on potential fields, is investigated and applied to station-keeping and diver-following. A deliberative approach, which plans the robot's motion over a finite horizon, is presented for diver leading. These approaches are applied to the SeaBotix vLBV300 platform for which a simulator is developed based on a decoupled motion model for the platform, as well as experimental results in a controlled test tank.

14. SUBJECT TERMS Unmanned Underwater Vehicle; Tethered; Hovering; Autonomous Underwater Vehicle; Joint human-robot operations; dynamic, uncertain environments			**15. NUMBER OF PAGES** 97
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

DIVER RELATIVE UUV NAVIGATION FOR JOINT HUMAN-ROBOT OPERATIONS

Andrew T. Streenan
Lieutenant, United States Navy
B.S.M.E., California Polytechnic State University, 2006

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2013

Author: Andrew T. Streenan

Approved by: Noel Du Toit
Thesis Advisor

Doug Horner
Thesis Second Reader

Knox T. Millsaps
Chair, Department of Mechanical and Aerospace Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A novel application for Autonomous Underwater Vehicles (AUVs) is considered here: a robotic diver assistant that enables close-quarters robotic operations with human divers. A robotic diver assistant has the potential to improve the efficiency, effectiveness and safety of diver operations. The robot diver assistant must share the operating environment with human divers, navigate relative to the environment to reach a specified site location (along with moving divers), and then maneuver among the mostly static divers as they perform their tasks on location. The robot operates in three unique scenarios: station-keeping, diver-following (shadowing), and diver-leading (vectoring). Various strategies for navigating among divers while ensuring diver safety are investigated. A reactive strategy, based on potential fields, is investigated and applied to station-keeping and diver-following. A deliberative approach, which plans the robot's motion over a finite horizon, is presented for diver leading. These approaches are applied to the SeaBotix vLBV300 platform for which a simulator is developed based on a decoupled motion model for the platform, as well as experimental results in a controlled test tank.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AUV	Autonomous Underwater Vehicle
CTE	Cross Track Error
DVL	Doppler Velocity Log
LOS	Line of Sight
PID	Proportional Integral Derivative
ROV	Remotely Operated Vehicle
THAUS	Tethered Hovering Autonomous Underwater System
UUV	Unmanned Underwater Vehicle

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Noel Du Toit for his guidance and expertise as my thesis advisor. Doug Horner's expertise in unmanned vehicles and guidance as a second reader was also a tremendous help.

I would especially like to thank my wife and son for their loving support throughout this process.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION

Traditional Autonomous Underwater Vehicle (AUV) research has focused on long-range, open-ocean operations. However, a fundamentally different application domain has received little attention: close-quarters operations. Importantly, this includes joint diver-robot operations. The underwater domain is inherently dangerous to humans, and diver operations are expensive due to limited diver bottom time and required surface support. A robotic diver assistant has the potential to improve the efficiency, effectiveness and safety of diver operations for all branches of professional divers (e.g., military, police, science, etc.). A core requirement for joint diver-robot operations is the ability to share the operating space with a human. This is the emphasis of this thesis. Of particular interest is the potential benefit to the Salvage, Explosive Ordinance Disposal (EOD), Undersea Rescue operations of the Department of Navy, and NASA (Figure 1).

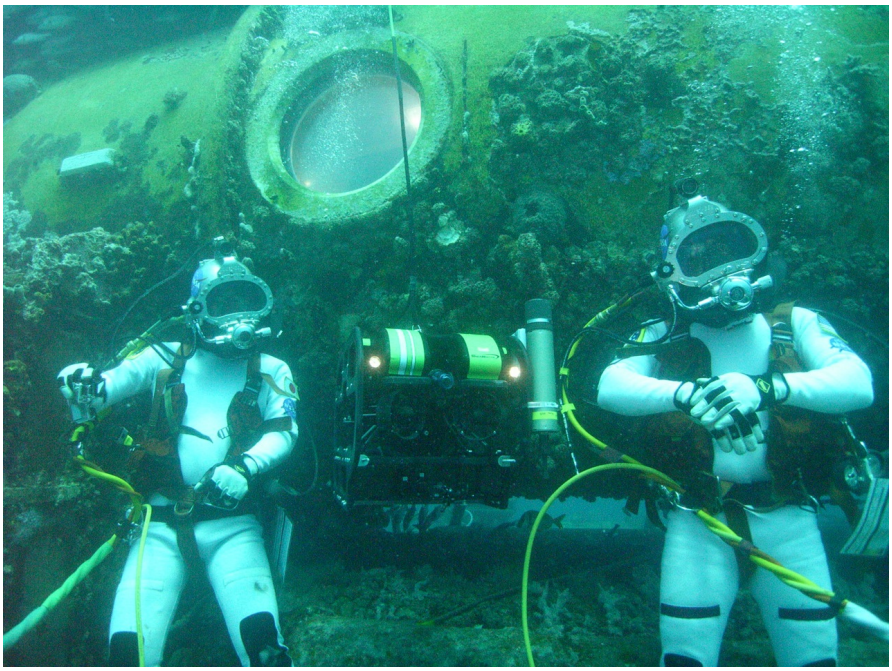

Figure 1. NASA divers in close-quarter operations with a UUV.

Underwater operations are challenging due to communication and vehicle power constraints, as well as the sensory-deprived nature of the underwater environment. However, one feature of the underwater domain that can be leveraged for joint human-robot operations is the slower environment dynamics: the divers' and robot's mobility is inherently constrained. These dynamic characteristics make this application domain attractive for the development of complex joint human-robot navigation algorithms since the decision cycle can be extended. On the other hand, accurate measurements of the underwater environment are difficult to obtain and large environmental disturbances exist. The robot has to operate in close proximity to the sea bottom (reefs, rocks, over-hangs, etc.) and divers and as a result it is critical to anticipate and account for the diver's motion, as well as disturbances and uncertainties in the environment when solving the planning problem to ensure diver and robot safety.

There are three distinct scenarios that are of interest in this work. A station-keeping scheme is required where the vehicle needs to travel to and stay in a particular position and orientation (aka pose). This pose does not necessarily have to be close to divers. Second, the vehicle needs to be able to track and follow a diver to support dive operations. Third, the vehicle needs to be able to plan and lead a diver from one location in the workspace to another. Various planning and execution approaches are available for close-quarter operations, and they can broadly be categorized as either reactive or deliberative approaches. Reactive approaches plan only one step ahead and thus do not take into account likely motion for operating with a diver. As such, the control of the robot can be less efficient during the execution of its mission, and more susceptible to environmental disturbances. Reactive approaches are also prone to local minima, but have the advantage that they are computationally efficient and simple to implement. Deliberative methods are able to account for anticipated motion and environmental disturbances since the planning problem is formulated over a longer planning horizon. They are less computationally efficient and more complicated to implement. The appropriate method strongly depends on the scenario of interest, and as a result both strategies are investigated here.

B. LITERATURE REVIEW

The primary objective for unmanned systems research is to eliminate the manned element from tasks that are “dirty, dangerous, or dull”. Traditional AUV research has focused on long-range, open-ocean type missions. Research related to close-quarter operations of AUVs and more specifically joint robot-diver operations, however, is limited. Two bodies of research that are related to joint human-robot interaction and close-quarter operations are the AQUA robot and the Hovering Autonomous Underwater Vehicle.

The AQUA robot [1], [2], shown in Figure 2, has been under development for a number of years, with a research emphasis on novel platform and propulsion design, as well as underwater computer vision techniques to detect and track divers and navigate in the underwater domain. Specifically, the robot uses a visual servo control to track and follow a human diver. The robot could then follow the same path on a future run. In contrast, this work focuses on control techniques to allow navigation among divers with a small, hovering class, tethered AUV that is additionally suitable for close quarters operations, and also considers more general interaction than just following divers, as mentioned above.

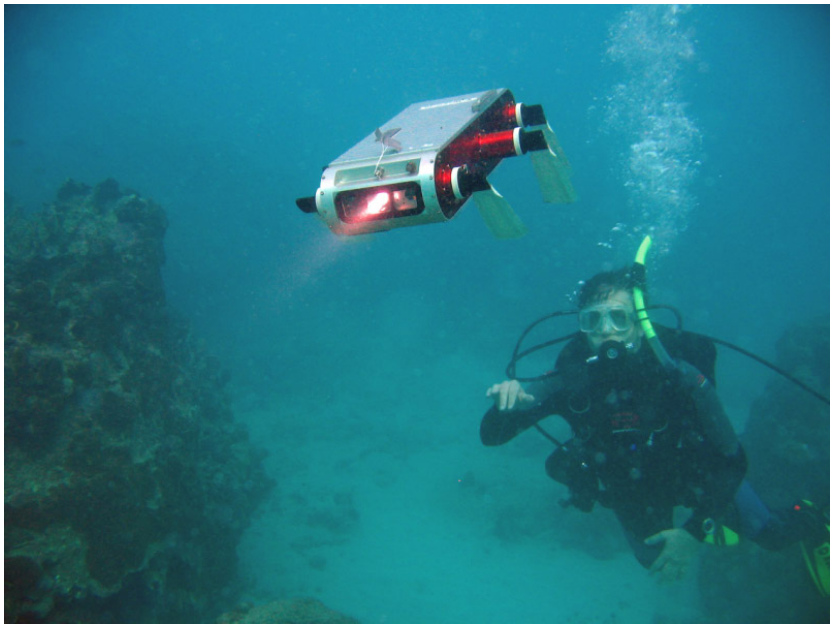

Figure 2. AQUA robot from [3]

Also related is the development of the Hovering Autonomous Underwater Vehicle (HAUV) to perform hull inspections. This medium-sized (82 kg) vehicle uses a Doppler Velocity Log (DVL) and sonar to navigate relative to and survey the hull of a ship [4]. These tasks require operation close to underwater objects, but the research is focused on the coverage problem: ensuring that they have mapped the entire hull. Joint robot-human operations are not investigated and the vehicle size largely disqualifies the vehicle for operation close to humans.

Diver relative navigation can be treated as formation control of an AUV relative to a diver. System theoretic and behavior-based methods for formation control, as applied to the underwater domain [5], attempt to mimic biological systems (e.g. bird formations). The system-theoretic approach [6] uses a decentralized method that uses a global center for the formation to track. In a decentralized method, each vehicle is controlled with onboard systems to track a global center of the formation (as opposed to a centralized method where a single system provides control to each vehicle). The vehicles attempt to maintain a pose relative to this global center and are less focused on relative distance to one another (i.e., each vehicle implicitly assumes all other entities behave as expected). Alternatively, Lewis and Tan propose a virtual structure [7] to maintain a geometric

relationship between multiple robots. As the robots move, this virtual structure is updated while corrections are applied to the robots to realign for formation control. These control approaches are generally applicable to a larger number of vehicles that can each be individually controlled. In contrast, this research is concerned with a small number of objects (including divers), of which only one can be controlled (i.e., the robot). Therefore, these methods are not considered here.

The three modes of operation of interest in this research are station keeping, diver following, and diver leading operations. Planning approaches based on reactive (e.g., potential field methods) and deliberative strategies (e.g., path planning and following), as applicable to joint robot-diver operations are investigated here. Reactive planners consider the current state of the environment and choose an appropriate action. As a result, these approaches do not exhibit anticipatory or predictive behavior. One popular reactive planning approach is known as Potential Field methods [8]. A potential field is a differentiable function, where the potential function itself can be thought of as energy, and it's derivative, or gradient, can be thought of as force. The gradient of the potential function at any configuration will yield the maximum rate of change of the potential function at that point. The negative of that gradient (gradient descent) applies a force to a particle to move the particle to a lower energy state in the field. Potential fields consist of the summation of attractive potentials (e.g., to guide the robot towards some goal) and repulsive potentials (e.g., to avoid obstacles) [8]. Potential field methods have been applied to establishing and maintaining formations of unmanned vehicles [9], avoiding obstacles [10], and changing the shape of the formation (e.g., [11]). These methods have also been applied to AUV obstacle avoidance [12]. A key benefit of these methods is the computational efficiency and the simplicity to implement various behaviors, but it is a purely reactive approach and does not predict the motion or activity of the diver. This can be problematic when large disturbances exist and can result in inefficient mission execution. The method is also prone to local minima in the energy function. Furthermore, since multiple potentials are summed to make up the overall behavior of the vehicle, the relative importance of each behavior must be specified. In practice it is challenging to define a weighting scheme that provides the desired behaviors for a large class of scenarios.

Deliberative strategies rely on some amount of a priori knowledge to be able to solve the trajectory-planning problem over some finite horizon: move the vehicle from some initial to a goal pose. In dynamic environments, the spatial and temporal planning problems must be addressed simultaneously (in general) in order to reach the goal location while avoiding static and dynamic obstacles. These approaches commonly rely on some form of optimization formulation, where the form is determined by the cost function of interest, the constraints in the system, and the form of the dynamic equations [13]. However, these optimization-based approaches tend to be computationally intensive. Other approaches to planning in dynamic environments include the Velocity Obstacles approach [14] and the Dynamic Window approach [15]. A final alternative is separating the spatial and temporal planning problems [13]. This approach is motivated by the fact that the only movers in the robot's environment are the divers. Thus, it is appropriate to plan the spatial path to avoid static obstacles in the environment while using temporal planning to avoid dynamic obstacles (i.e., divers). This is similar to human operations with cars. This separated spatial and temporal planning approach tends to be computationally more efficient, and is pursued here.

Spatial path planning algorithms are again commonly formulated as optimization problems, but these approaches tend to be computationally intensive. Alternatively, the solution space can be discretized (i.e., a graph-based approach) to efficiently obtain near-optimal solutions. Graph-based approaches include methods such as visibility graphs, Voronoi diagrams, and probabilistic roadmaps [8]. Planning spatial paths inherently requires substantial a priori information to construct the graphs and are often only defined for static environments. These algorithms normally consist of a graph generation/update step and a graph search step. In static environments, these graphs can be generated once and the planning process can become very efficient since it only consists of a graph-search step. Various graph searching methods can be employed to this end (e.g., depth first search, breadth first search, A-star) [8]. These searches attempt to find the lowest cost path in the graph to connect the goal location to the initial location. A popular method for discretizing the problem space (i.e., generating the graph) is the so-called incremental sampling approach. These graphs are incrementally generated by randomly

selecting configurations and connecting these configurations back into the graph while accounting for obstacles and other constraints (including the dynamics of the vehicle). Trees are a special type of graph where each node is only allowed to have a single parent. This saves computational time since the search starts at the goal and goes to each parent until the initial point is reached [8]. Examples of incremental sampling-based planning algorithms include Rapidly-exploring Random Trees (RRT) and Expansive Space Trees (EST) [8]. These methods are *feasible* planners: a feasible path is obtained (potentially in very complex and high-dimensional state spaces). A version of RRT, known as RRT*, has recently been proposed [16]. This algorithm additionally rewires the tree upon node generation according to proximal node costs and has been proven to be asymptotically optimal. This method is investigated further in this work.

C. OBJECTIVES

The focus of this research is the development of a planning and control strategy that allows joint robot-human diver operations with an Unmanned Underwater Vehicle. An approach is desired that can simultaneously avoid the divers to ensure safety and follow or lead the divers to provide mission assistance. The objectives are:

- Perform station-keeping operations without any interaction with divers.
- Follow a diver while maintaining a specified relative pose.
- Plan a path and lead a diver to a specified location while avoiding obstacles.
- Demonstrate these operations in simulation as well as experimentally (in a controlled dive tank environment).

Based on these objectives, the following research tasks have been identified:

- Develop a decoupled model of the development platform, which can be used for planner/controller development and as a basic simulation capability.
- Investigate planning strategies (e.g., reactive and deliberative methods) that allow the vehicle to follow a diver, generate a path, and lead a diver to a goal location while avoiding obstacles and ensuring diver safety.
- Develop executive control strategies (in particular the path following controller) that allow the vehicle to follow a generated path with minimal deviation.

THIS PAGE INTENTIONALLY LEFT BLANK

II. EXPERIMENTAL SETUP

A. SEABOTIX VLBV300 PLATFORM

To operate in close-quarters with humans, the platform must be small, agile, and be able to respond in multiple directions, including sway, to ensure diver safety. The platform should have sufficient control authority to overcome currents, surge, and other environmental disturbances and must allow for the integration of various sensor payloads, including sonar, a Doppler Velocity Log (DVL), and Inertial Navigation System (INS).

The experimental platform for this research is the SeaBotix vLBV300 platform, shown in Figure 3 [17]. It is a tethered, Remotely Operated Vehicle (ROV). The vLBV300 has six thrusters, two for vertical and roll control, and four vectored thrusters for control in the surge, sway, and yaw directions, as shown in Figure 3. A computer control-interface has been developed for the vehicle that leverages a high-level (joystick) control interface (surge, sway, heave, yaw) and individual thruster control (surge, sway, heave, yaw, roll), resulting in a Tethered, Hovering Autonomous Underwater System (THAUS).

Figure 3. The SeaBotix vLBV300 tethered AUV platform (*left*), and the planar vectored thruster configuration (*right*) from [18].

The vehicle coordinates are chosen to be positive *x* in the forward surge direction, positive *y* in the starboard sway direction, and positive *z* in the downward heave direction (refer to Figure 4). Yaw is defined between $-\pi$ and $+\pi$ with zero in the positive *x* direction and increasing clockwise.

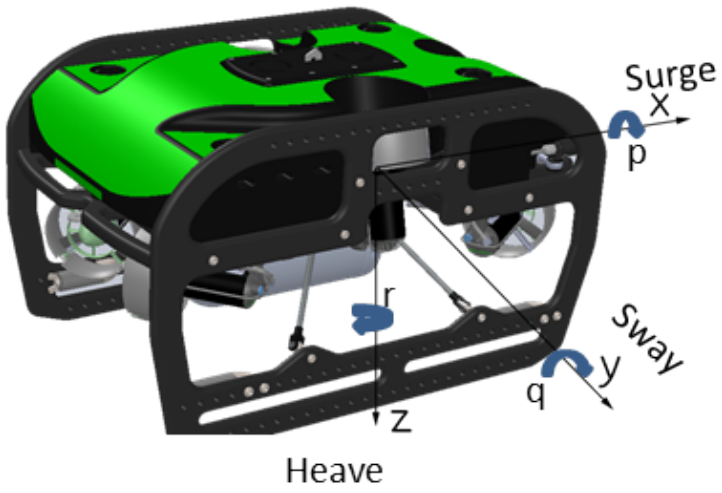

Figure 4. THAUS body reference frame and variable definitions after [18].

For this research, the platform has no onboard sensors (e.g., an inertial navigation system) and position and orientation measurements are obtained from an external system, which is discussed next.

B. VEHICLE LOCALIZATION AND TRACKING

The vehicle position and orientation must be known at all times in the operating space to allow for autonomous control, verifying system response and ensuring diver safety. Ideally, internal sensors (e.g., INS, camera, sonar, DVL, etc.) would supply this information. However, these are not currently available and this is a notoriously hard problem to solve. As a result, position and orientation measurements are obtained from an external localization system in this research.

Position and orientation (aka pose) data for the vehicle are determined using a VICON motion capture (MoCap) system. The infrared (IR) camera-based MoCap system

provides accurate marker tracking (<1cm) at a high rate (100 Hz) [19]. However, due to electro-magnetic wave attenuation in water, the current MoCap setup can only be utilized above the water surface. As a result, the THAUS has been extended with a low-inertia structure above the water surface, which can be tracked by the MoCap and from which the vehicle pose can be calculated. This structure does affect the vehicle dynamics, but these effects are neglected. Four infrared cameras surround the experimental test tank (see Figure 5).

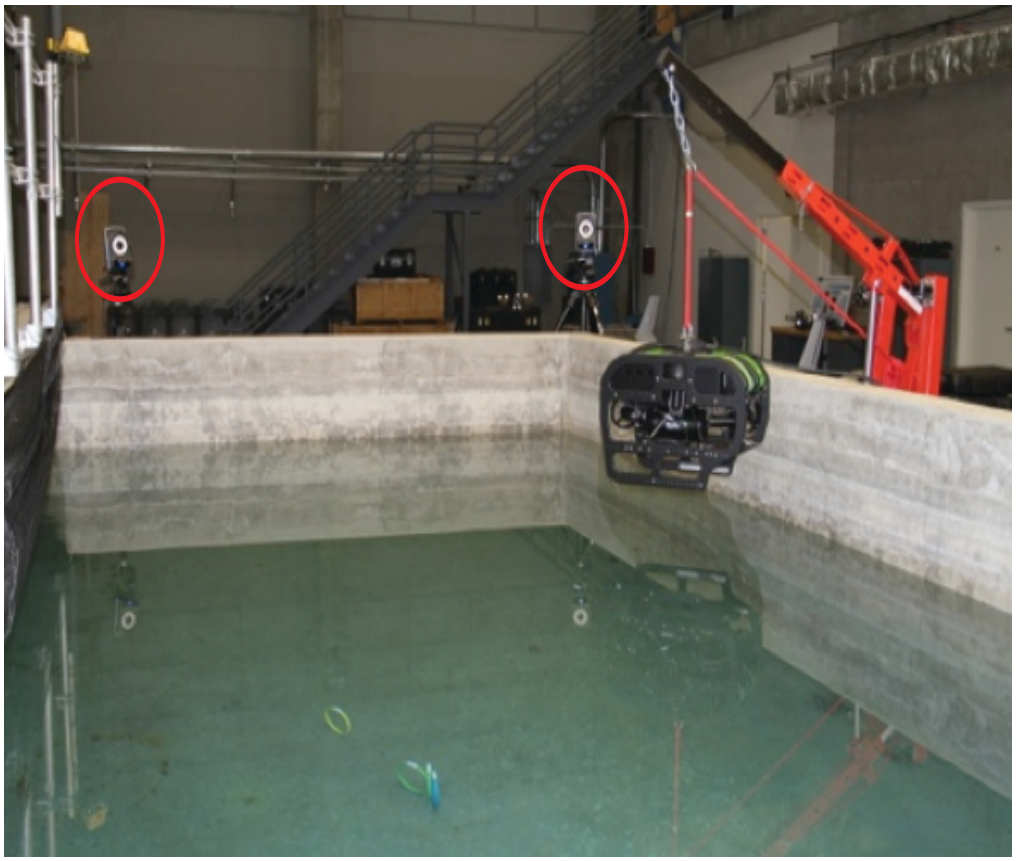

Figure 5. NPS instrumented dive tank (VICON cameras circled in red).

The default coordinate system provided by the MoCap setup is a right handed, north, east, up, or NEU (for *x*, *y*, and *z*, respectively), coordinate system. For UUVs, the coordinate system of interest is north, east, and down, or NED (for *x*, *y*, and *z*, respectively). Therefore, the default coordinate system is transformed to the NED system by applying a negative gain to the yaw orientation to achieve positive rotation clockwise.

Additionally, the NED coordinate system is rotated into a body fixed frame of reference by the following rotation matrix:

$$R = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \tag{0}$$

where ψ is the yaw angle of the vehicle.

This localization information is used to develop a simple, decoupled motion model for the vehicle, as well as implement and execute planning and control strategies. The dive tank's approximate dimensions are 6 meters by 3 meters by 2 meters deep. The use of this setup effectively limits the experimental setup to a two-dimensional, three-degree of freedom system (i.e., planar motion): surge, sway, and yaw. The *z* direction is not investigated due to the limiting space. Additionally, obstacle avoidance for deliberative methods is only simulated due to the space constraints for experimental runs.

C. DIVER DETECTION AND TRACKING

Diver detection and tracking is an important and challenging research problem. Ideally, diver detection and tracking will be performed on sensors onboard the THAUS, using both electro-optical and acoustic data. However, this is research topic in itself and falls outside the scope of this thesis. Instead, a virtual diver is used during development of the control and navigation strategies to demonstrate the joint robot-diver capabilities. It is assumed that the position and orientation of the diver is known, and a simple, constant velocity dynamic model of the diver is assumed (if necessary).

III. DECOUPLED DYNAMIC MODELS

A detailed, 6-DOF hydro-dynamic model of THAUS is currently being developed [20]. However, a simplified, decoupled dynamic model is desired to capture simple motion for the vehicle in the surge, sway, and yaw directions (i.e., planar operations) for this work. First, the response of the system to high-level commands is investigated, before extending this model to allow position control. Based on these models, a basic simulation capability is developed for planar motion. The models and associated assumptions are presented below. The Simulink block diagrams and associated MATLAB scripts for the simulator are contained in Appendix A.

A. VELOCITY RESPONSE

Utilizing standard control theoretic model identification techniques (e.g., see [21]), the decoupled planar dynamics of THAUS are determined by specifying a step input in the surge, sway, and yaw directions, respectively, while recording the vehicle response in the test tank using the VICON Motion Capture system. The high-level (joystick) interface is used. These commands are scaled thruster RPM commands, resulting in a specified thrust. This relationship between the high-level command and generated thrust is non-linear in general, but it is implicitly assumed that the mapping between the joystick command and the resulting force is linear for the simplified, decoupled models of interest here. Let *v(t)* be the velocity and *f(t)* be the force input (i.e., joystick command). In general, the transfer function for the plant is defined as (for surge, sway, and yaw):

$$P(s) = \frac{V(s)}{F(s)} \tag{1}$$

where *V(s)* and *F(s)* are the Laplace transforms of *v(t)* and *f(t)*, respectively (see Figure 6). *f(t)* is a step input of magnitude K_j, resulting in the following transfer function:

$$F(s) = K_j \frac{1}{s} \tag{2}$$

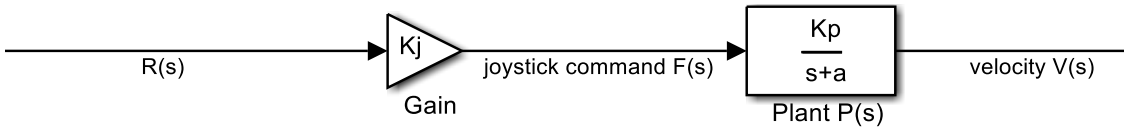

Figure 6. Block diagram of the open-loop system for a single channel (i.e., decoupled), assuming a first-order system response.

Based on the response of the system in each individual channel (e.g., see Figure 7 for the surge direction), a first-order model is assumed for the plant, written in terms of the steady state velocity (v_f) and time constant (t_c). The transfer function for a generic first order system is:

$$P_F(s) = \frac{K_p}{(s + \frac{1}{t_c})} \tag{3}$$

where $K_p = \frac{v_f}{K_j}$ can be related to the step-response variables.

As an example, the step input for the model is compared with the experimental results in the surge direction in Figure 7. This formulation implicitly assumes that the mapping between the joystick command magnitude (K_j) and the generated thruster force is linear. It is also assumed that the model will operate close to the trim condition for which it is developed. This is not strictly the case, but since the objective is to develop simplified, decoupled dynamic models that will be used with feedback controllers, the resulting models are sufficient. As can be seen in Figure 7, the velocity appears to be increasing slightly at the end, but longer datasets are not possible due to the experimental setup. Recommendations for overcoming this limitation in the experimental setup is presented in the Conclusions section.

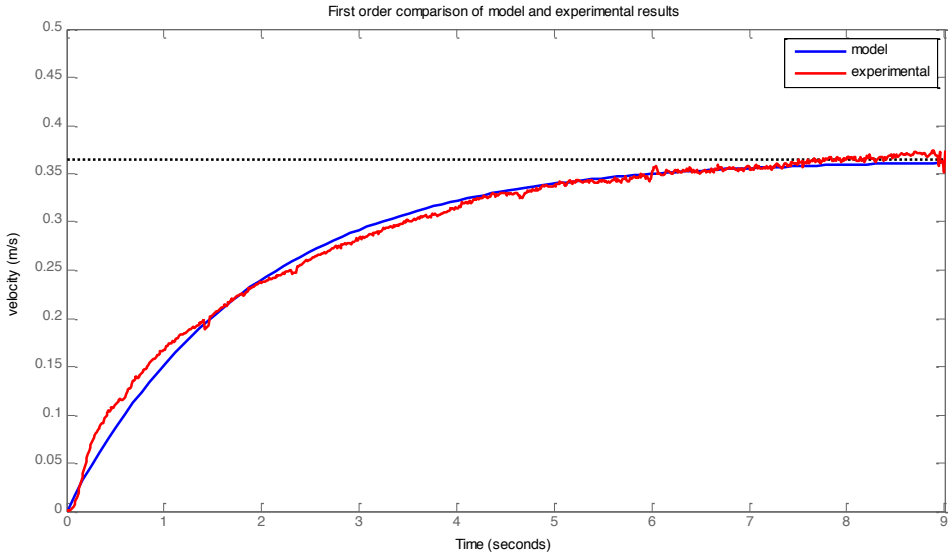

Figure 7. Comparison of velocity step responses from vehicle and model in the surge direction. Similar results were obtained for the sway and yaw channels.

The resulting 1st order systems for the independent channels are presented in Table 1. Note that the assumption of decoupled dynamics suggests that these behaviors can be superimposed when coupled motions are executed. The validity of this assumption will be addressed below.

Channel	Transfer Function
Surge	$\frac{0.000622}{s+0.769}$
Sway	$\frac{0.000267}{s+0.667}$
Yaw	$\frac{0.0045}{s+2.5}$

Table 1. First order transfer functions determined from velocity response in the individual channels

From Table 1 it is clear that the yaw time constant is the smallest, and therefore has the fastest dynamics of the three channels. In practice this means that yaw corrections can be made more readily than the other channels. Furthermore, the sway response has

the largest time constant (slowest dynamics), which makes sense since the vehicle has the least amount of control authority in this direction.

B. POSITION RESPONSE

For the applications of interest, it is additionally desirable to control the position of the system. The first order (velocity) system generated in Section III.A can be integrated to obtain position. This position data can then be used with negative feedback control to achieve a desired position (see Figure 8). With the first order (velocity) system determination, a constant force was used, and the resulting velocity response was assumed to be linear. An alternative method to approximately account for the non-linearities in the system response is to apply a varying magnitude force. The same setup in Figure 8 can be used to measure the position response and to reevaluate the values for K_p and a. In order to develop a simple, decoupled model for the position step response, it is assumed that the required command (and thus thrust) is proportional to the position error between the reference and the actual vehicle position.

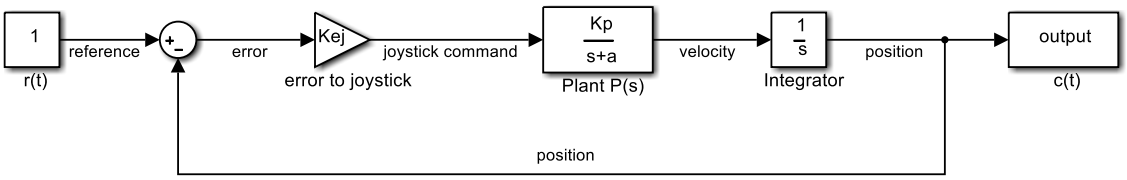

Figure 8. Block diagram of closed loop system resulting in second order response for position control.

A generic second order transfer function is given by:

$$G(s) = \frac{\omega_n^2}{s^2 + 2 \cdot \zeta \cdot \omega_n + \omega_n^2} \tag{4}$$

From the step-input response of the second-order dynamic system, the values for percent overshoot and peak time can be used to completely define a second order system:

$$\zeta = \frac{-\ln(\%OS/100)}{\sqrt{\pi^2 + \ln^2(\%OS/100)}} \tag{5}$$

$$\omega_n = \frac{\pi}{T_p\sqrt{1-\zeta^2}} \tag{6}$$

The values that are determined are for the second order position response of the closed loop system (see Figure 8). The open loop and closed loop system are related by

$$G_e(s) = \frac{G(s)}{1+G(s)} \tag{7}$$

where G(s) is the open loop transfer function. The open loop transfer function can then be calculated as:

$$G(s) = \frac{G_e(s)}{1-G_e(s)} \tag{8}$$

From the block diagram in Figure 8:

$$G(s) = K_{ej}P(s)\frac{1}{s} \tag{9}$$

and therefore the plant can be determined from the measured closed-loop response:

$$P(s) = \frac{G(s)s}{K_{ej}} \tag{10}$$

where K_{ej} is the gain that scales the position error to force. Substituting the measured values for ζ and ω_v as well as the constant K_{ej}, the transfer function for the plant is :

$$P(s) = \frac{\omega_n^{\,2} / K_{ej}}{s + 2\zeta\omega_n / K_{ej}} \tag{11}$$

Figure 9 shows a comparison of a step response from the model determined by ζ and w_n to the actual response (for example in the surge direction). The model initially follows the vehicle response up to the maximum overshoot. The model and plant response then diverge due to the un-modeled nonlinearity in the joystick-force relationship. The settling time of both responses is about the same however, so this deviation is acceptable as the initial transient and final steady state values are the same. The resulting 1st order transfer functions for the plant in the surge, sway, and yaw directions, based on this approach, are given in Table 2.

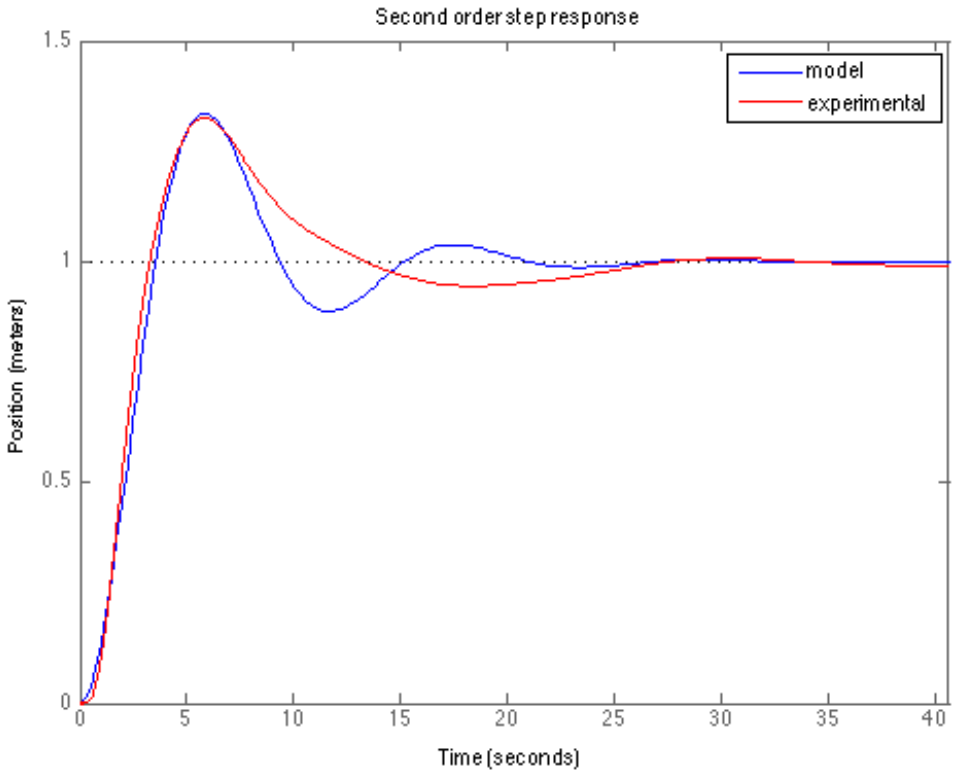

Figure 9. Comparison of position step responses from vehicle and model in the surge direction. Similar responses are obtained for the sway and yaw channels.

Channel	Transfer Function
Surge	$\frac{0.000246}{s+0.238}$
Sway	$\frac{0.000126}{s+0.217}$
Yaw	$\frac{0.00381}{s+0.714}$

Table 2. Transfer functions based on position response of the system in the individual channels

C. SIMPLIFIED, DECOUPLED MODEL COMPARISON

In order to compare the responses for the velocity and position response models, the first order velocity model is used as part of a position controller (i.e., in the closed loop transfer function as in Figure 8) to demonstrate the difference in behavior when

subjected to a step input in reference position. Figure 10 shows the comparison of the actual plant response in addition to the position and velocity models in the surge direction of the vehicle.

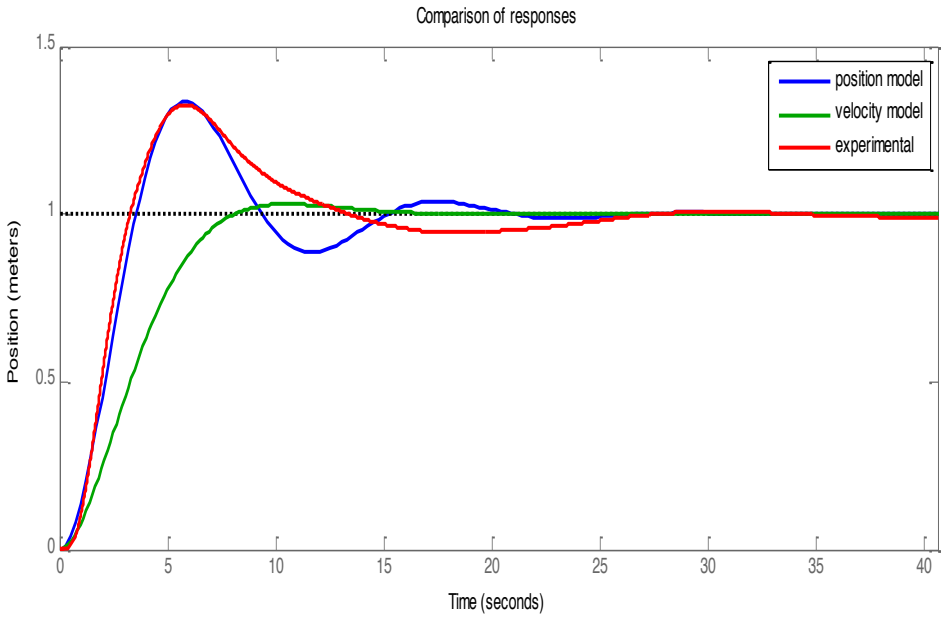

Figure 10. Comparison of velocity and position models and the experimental results for a step input in position.

Both models reach a steady state value of one. The velocity model shows a slower response time and a higher damping ratio than the position model. The position model has a similar rise slope, peak overshoot, and settling time as the actual plant. From this result it is concluded that the position response model measured from the assumed 2nd order dynamics more accurately captures the system response and is used for plant modeling and control determination in the following section.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. TECHNICAL APPROACH

For this research, the THAUS is to operate in three separate scenarios:

- station-keeping, where THAUS moves towards and maintains a specified pose without regard to a diver;
- diver-following, where THAUS tracks and follows a diver;
- diver-leading, where THAUS leads a diver to a goal location.

These modes of operations have distinct requirements for planning, control, and navigation and the developed approaches in each case are presented below. A PID controller and potential field are proposed for both station-keeping and diver-following operations. A potential field and deliberative planning method are proposed for diver-leading operations. The Simulink block diagrams and MATLAB scripts associated with these modes are contained in Appendix B.

A. PID CONTROL

One of the fundamental requirements for underwater operations is to move into, and maintain, a specified pose while minimizing offset errors and rejecting disturbances. PID controllers are commonly used for this purpose. Using the simplified, decoupled models of the system developed for the surge, sway, and yaw directions, three proportional, integer, and derivative (PID) controllers are implemented for the surge, sway, and yaw channels (see Figure 11).

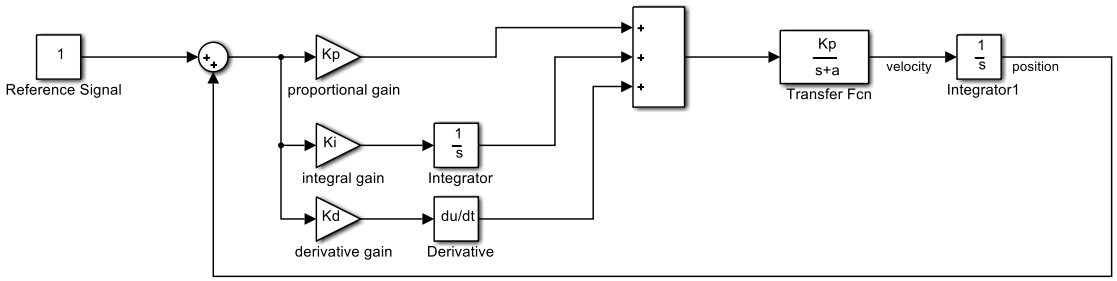

Figure 11. Block diagram of PID controller applied to transfer function

The design criteria for each channel include: less than 10% overshoot, less than 20 seconds settling time, and 0 steady-state error. The uncompensated root locus plot and corresponding step response for the surge channel are shown in Figures 12-13.

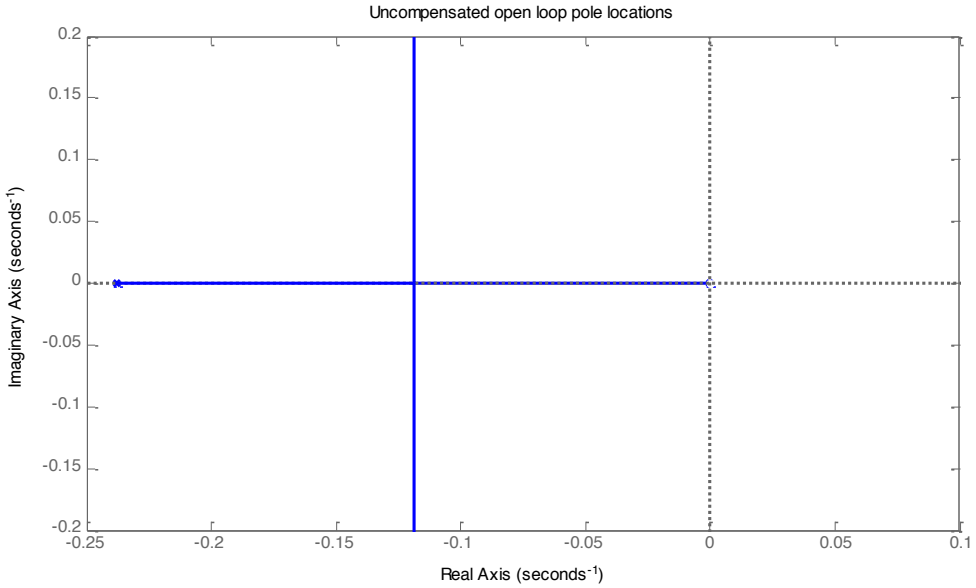

Figure 12. Uncompensated, open loop pole locations and branches for surge motion.

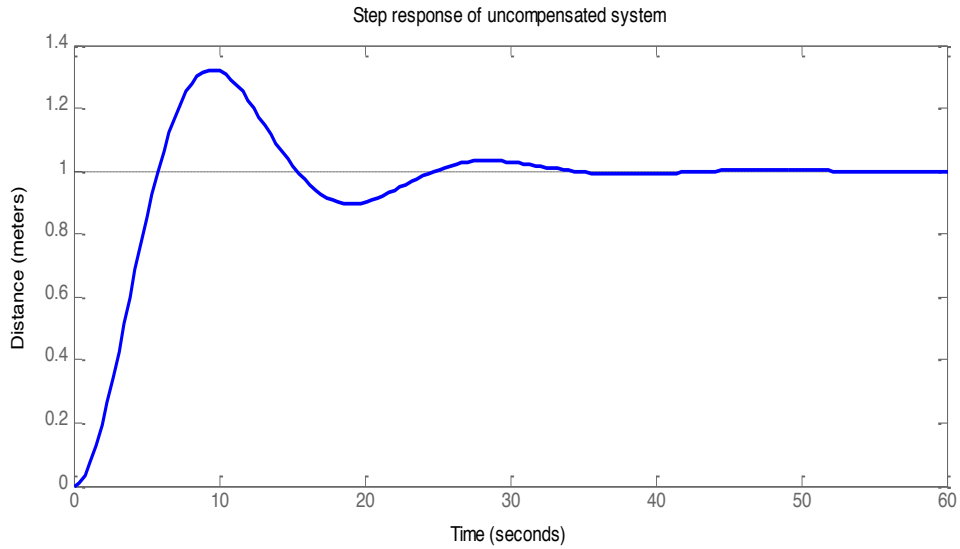

Figure 13. Uncompensated step response in surge motion.

The settling time for the uncompensated system is over 20 seconds with 30% overshoot. The root locus only has two branches that depart at a 90-degree angle. While the response time can be lessened by just adding a proportional gain along those branches, further design criteria like damping ratio, settling time, and peak overshoot cannot be adjusted. To make these adjustments, integral- and derivative feedback control are added and tuned to provide the desired response. Figures 14 and 15 show the updated root locus plot and corresponding, closed loop step response for the compensated system.

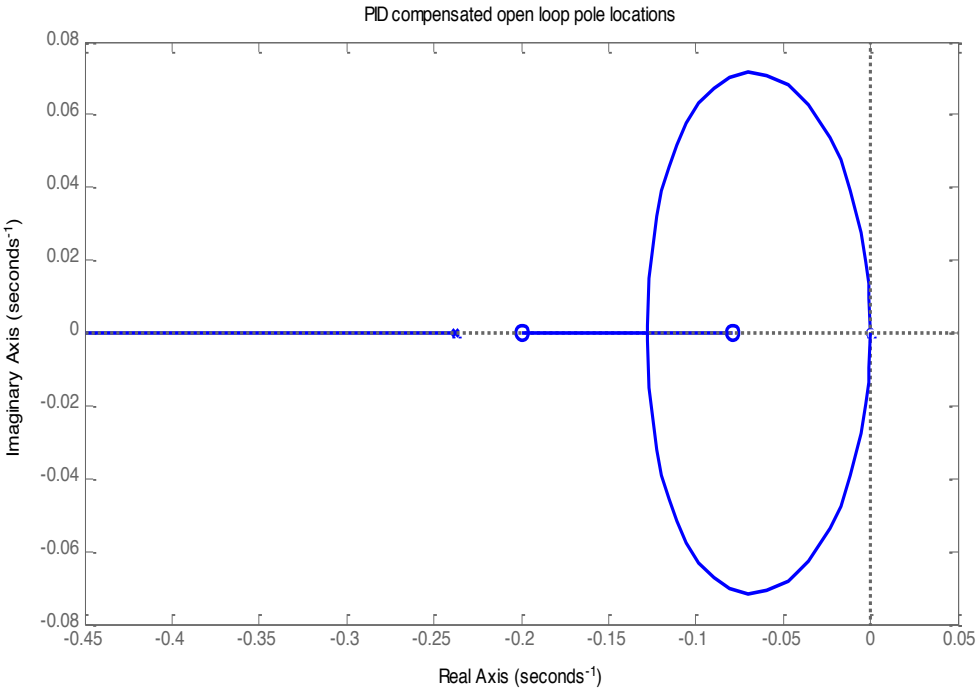

Figure 14. Compensated, open loop pole locations and branches for surge motion.

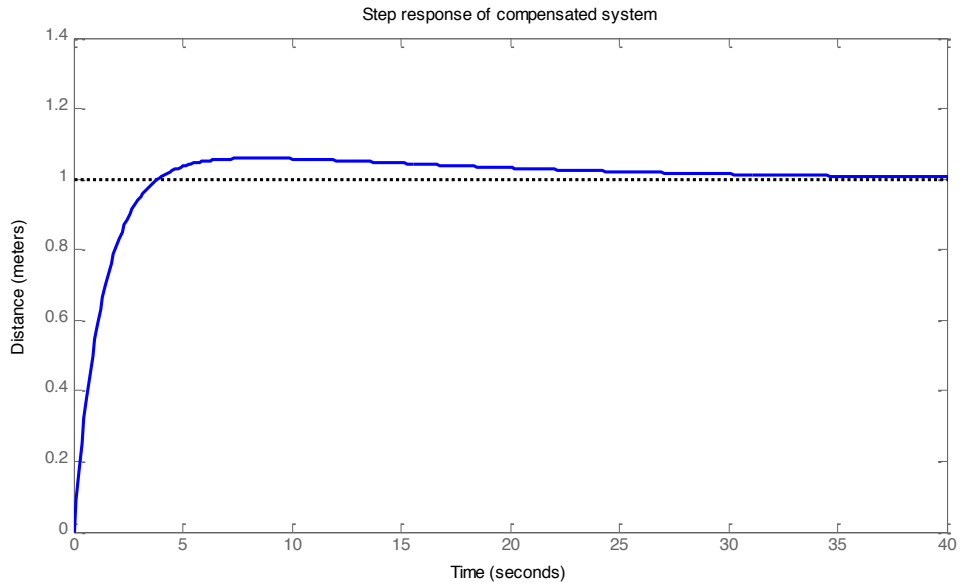

Figure 15. Compensated step response in surge direction.

The PID controller adds additional branches to the root locus for tuning, and the step response shows the desired criteria of less than 10% overshoot, less than 20 seconds settling time, and 0 steady-state error. Similar results for the sway and yaw channels are obtained. Table 3 contains the values for the PID controllers for each channel.

Channel	Proportional	Integral	Derivative
Surge	885	10	3175
Sway	1500	10	6000
Yaw	315	4	320

Table 3. PID controller design for station keeping

The greatest limitation of PID control for close-quarter operation is the inability to use it to avoid obstacles. The controller simply looks at the distance between the THAUS and the goal to calculate the control signal. Additionally, large errors can cause integral windup, leading to a vehicle moving too quickly towards a diver to ensure safety. These limitations motivate the potential field approach, which is discussed next.

B. POTENTIAL FIELD APPROACH

A diver may need to perform work underwater: hull inspection, underwater mine neutralization, or underwater archeology. Sharing the workspace with divers require both station-keeping and diver avoidance. The diver is in charge of the workspace and the robot must respond to diver motion. Thus, the robot must maintain a pose while additionally avoiding obstacles in the environment. Artificial potential fields are useful for this purpose since the THAUS will simply be reacting to and following the diver's movements.

A potential field is a differentiable function, where the potential function itself can be thought of as energy, and it's derivative, or gradient, can be thought of as force [8]. The gradient of the potential function at any configuration will yield the maximum rate of change of the potential function at that point. The negative of that gradient (gradient descent) defines a force that, when applied to a particle, moves the particle to a lower energy state in the field.

Potential field methods are attractive due to their computational efficiency and implementation simplicity. Various behaviors can be individually implemented as discrete potentials, the effects of which can then be super-imposed to obtain more complex behaviors. Potential fields consist of the sum of attractive potentials (e.g., to guide the robot towards some goal) and repulsive potentials (e.g., to avoid obstacles).

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{12}$$

The negative of the gradient of the potential function defines a force on a particle. This particle is assumed to be able to move omnidirectionally along the gradient. Particle motion does not account for vehicle dynamics (turn rate, pose, etc.), but since THAUS has three-degrees of freedom in the horizontal plane, particle motion is a reasonable assumption.

One drawback of the method is that the relative importance of these behaviors must be defined (by scaling the effects of the individual potentials). It can be challenging to get a single relative scaling that is appropriate in all situations.

In this work, the position and orientation of the diver is assumed to be known and the THAUS' goal is specified relative to this diver's location (see Figure 16): 1 meter to the diver's port side and pointing to a position 1 meter ahead of the diver (to simulate illuminating the diver's workspace). Note that diver detection and tracking based on onboard sensors are beyond the scope of this thesis.

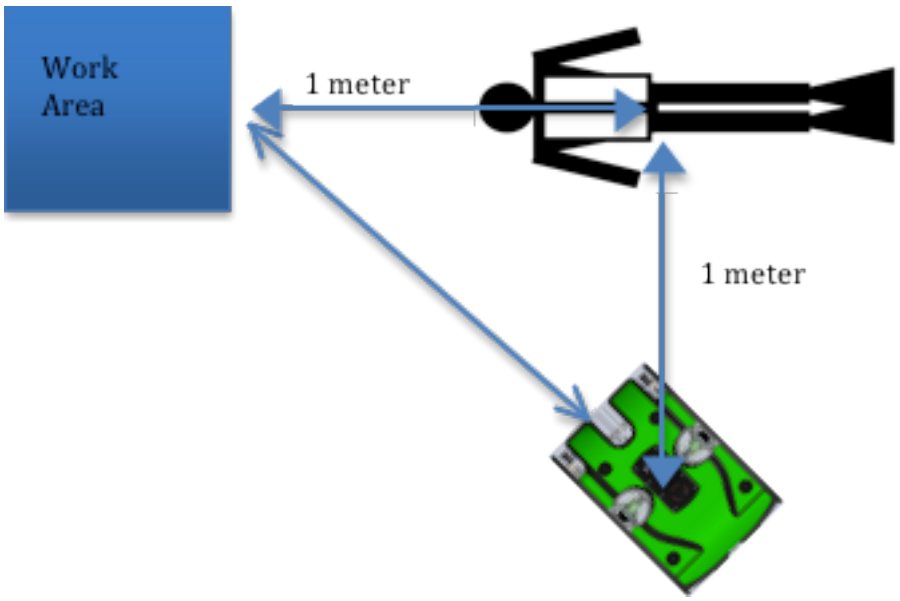

Figure 16. Desired pose for the THAUS in relation to the diver.

1. Attractive Potentials

One choice for the attractive potential to draw the vehicle to a goal location is to make the gradient of the potential proportional to the distance between the THAUS and the goal. As the THAUS travels towards the goal, the generated force of the attractive potential will decrease. The attractive potential given by [8] is used to pull the THAUS towards the goal location (refer to Figure 17):

$$U_{att} = \frac{1}{2}\zeta d^2(q, q_{goal}) \tag{13}$$

whose gradient is

$$\nabla U_{att}(q) = \zeta(q - q_{goal}) \tag{14}$$

ζ is a three-element proportional constant (scaling factor), used for each channel to achieve satisfactory response, q is the pose (position and orientation) of the THAUS, q_{goal} is the goal location and d is the distance between the THAUS and the goal. This attractive gradient is essentially a proportional controller that uses the distance between the THAUS and the goal as the error signal with a proportional constant to minimize the error. As a result, the potential field method can be used for station keeping as well. As

illustrated in Figure 17, the gradient (and thus the attractive force) is linear, and the gradient is zero at a distance of zero.

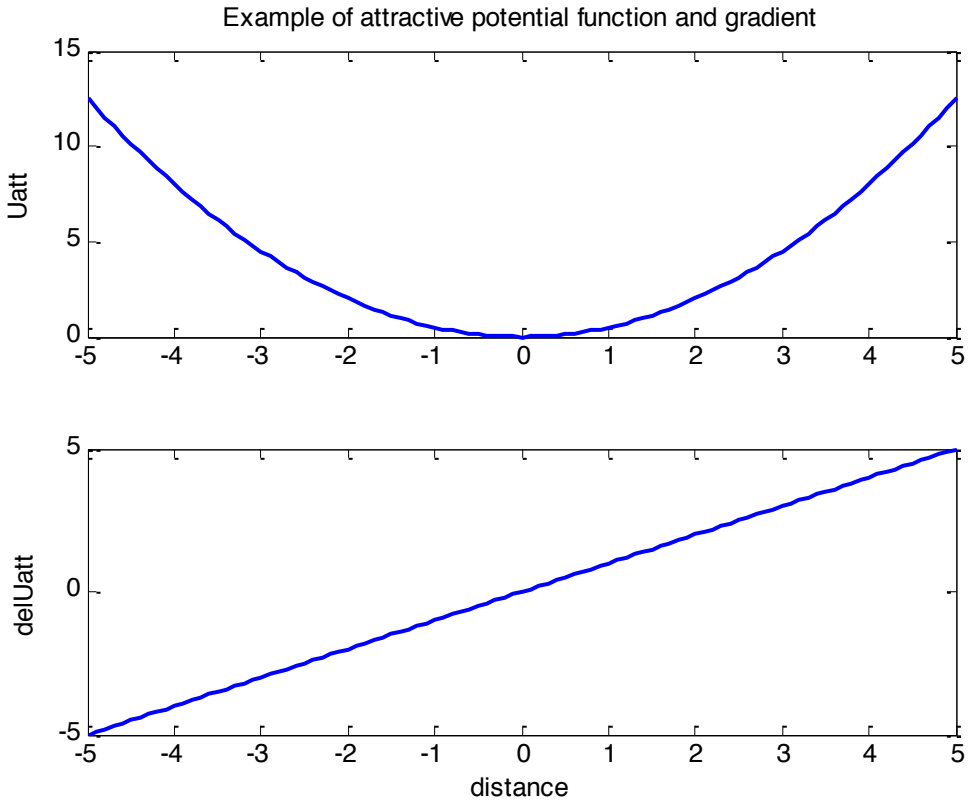

Figure 17. Attractive potential (top) and gradient (bottom) that defines an attractive force towards the goal location.

2. Repulsive Potential

The repulsive potential is used to keep THAUS away from obstacles (including divers). As the distance between the THAUS and the diver decreases, the gradient of the potential must increase. To ensure diver safety, an aggressive repulsive potential is used to push the robot away from the diver (refer to Figure 18). As the distance between the robot and diver approaches zero (i.e., a collision), the repulsive force approaches infinity:

$$U_{rep}(q,q_{obs}) = \frac{1}{2}\eta(\frac{1}{d(q,q_{obs})} - \frac{1}{Q^*})^2, d(q,q_{obs}) \le Q^*$$
$$U_{rep}(q,q_{obs}) = 0, d(q,q_{goal}) > Q^*$$

whose gradient is

$$\nabla U_{rep}(q,q_{obs}) = \eta(\frac{1}{Q^*} - \frac{1}{d(q,q_{goal})})\frac{1}{d^2(q,q_{goal})}\nabla d(q,q_{goal}), d(q,q_{goal}) \le Q^*$$
$$\nabla U_{rep}(q,q_{goal}) = 0, d(q,q_{goal}) > Q^* \quad (15)$$

d is the distance between the THAUS and the obstacle (diver), η is a proportional constant to achieve desired results, and Q^* is a threshold distance beyond which this behavior is ignored (i.e., the THAUS is far from the diver). From Figure 18, the function yields zero gradient (and conversely force) far away from an obstacle, but the gradient (force) quickly increases as the distance approaches zero.

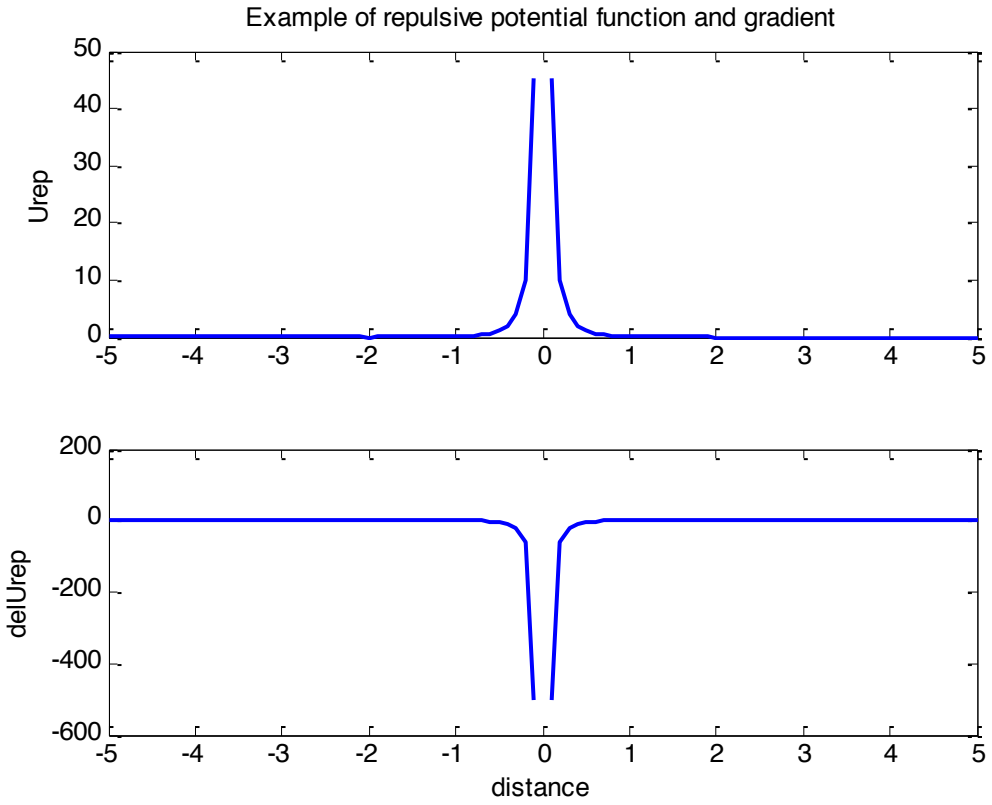

Figure 18. Repulsive potential (top) and gradient (bottom) that defines a repulsive force to maneuver the THAUS away from a diver

As an additional measure of safety, at a minimum safety circle from the diver the attractive potential is disabled.

The limitation of the potential field method is that it is purely reactive. This method does not anticipate diver motion and does not reason about global path planning (i.e., through an obstacle field). This method will be inadequate when attempting to lead divers along a path with several obstacles.

C. DELIBERATIVE PLANNING

For the final mode of operation, it is not only the THAUS that is reactive but also the diver. The THAUS now leads the diver from one location to another. For this mode of operation, the THAUS is assumed to only maneuver in the horizontal plane, with a known start and goal location. Additionally the environment is assumed to be static, except for the divers, and reasonable *a priori* information of the configuration space is available. This portion is broken into three sections: a path-planning algorithm, a velocity scaling algorithm along the spatial path, and path following controller.

Incomplete environment information introduces uncertainty into the planning problem. The recommended approach for handling this uncertainty due to incomplete environment state information is to frame the problem in terms of Receding Horizon Control: the planning problem is solved over some finite horizon, but only a fraction of the plan is executed before the entire planning problem is re-solved to account for new information. In this framework, the trajectory-planning algorithm used needs to be computationally efficient to allow re-planning as new information (obstacles) become available. Furthermore, the approach needs to be near-optimal to ensure efficient motion. This requirement for near optimality is motivated by the energy intensity of divers' motion in the underwater environment.

Since the environment is assumed to be mostly static with only other divers moving in the environment, it is reasonable to separate spatial and temporal planning [13], [16]. An analogous example of separating spatial (i.e., *path* planning) and temporal planning is car driving: spatial planning is used to satisfy traffic rules, follow roads, and avoid static obstacles, while temporal planning is used to avoid other vehicles.

1. Spatial (Path) Planning

The goal for many path-planning algorithms is to find a path between an initial starting location and a goal location. Spatial path planning algorithms are commonly formulated as optimization problems, but these approaches tend to be computationally intensive. Alternatively, the solution space can be discretized to obtain near-optimal solutions, including roadmap methods (e.g., visibility graph, Voronoi diagram, probabilistic roadmaps). These algorithms normally consist of a graph generation/update step and a graph search step. In static environments, these graphs can be generated once and the planning process can become very efficient (i.e., multi-query approaches). Once the graph is created, different search routines can be used to find a path between the initial and goal location (e.g., breadth first search, depth first search, A-star). If sufficient *a priori* knowledge of the environment is available, a single path can be created and followed. If the *a priori* information is incomplete, or the robot operates in a dynamic environment, the path will have to be re-planned to account for new information and environmental changes.

Trees are a special type of graph where each node is only allowed to have a single parent. This saves computational time since the cost evaluation is built into the tree structure during generation and the search starts at the goal and progresses up the tree to the initial location. Rapidly exploring random trees (RRT), RRT* and expansive-spaces trees (EST) are examples of trees.

For this implementation, a variant of the RRT method is used to plan a spatial path. RRT can account for vehicle dynamics and plan a path that is *feasible* within vehicle constraints, but the path may not be efficient. For the THAUS, the vehicle can operate independently in surge, sway, and yaw, so accounting for vehicle dynamics is not strictly necessary for path planning. The tree is defined by vertices, V, and edges, E, that defines the connections between the vertices. Vertices commonly correspond to poses in the configuration space.

In RRT, the graph is generated as follows: a random point is generated, $x_{rand,}$ in the configuration space. The *Nearest* function then finds the nearest vertex in the existing

tree, $x_{nearest}$. The *Steer* function then steers $x_{nearest}$ towards x_{rand} in a straight line generating a new point, x_{new}. The *ObstacleFree* function determines whether a collision occurs on the path connecting $x_{nearest}$ and x_{new}. If not, x_{new} is added to V, and the edge between $x_{nearest}$ and x_{new} is added to E. This continues until x_{new} equals q_{rand} or x_{new} encounters an obstacle. Next, the algorithm then searches from the goal location to the nearest point on the tree and determines if a collision free path exists. If a path exists the algorithm terminates, if not, a new random point is generated and the algorithm continues. The algorithm, adapted from [16], is shown in Figure 19.

1: $V \leftarrow \{x_{init}\}$; $E \leftarrow 0$;
2: for i = 1,...,n do
3: $x_{rand} \leftarrow SampleFree_i$;
4: $x_{nearest} \leftarrow Nearest(G = (V, E), x_{rand})$;
5: $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$;
6: if ObstacleFree($x_{nearest}, x_{new}$) then
7: $V \leftarrow V \cup \{x_{new}\}; E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$;
8: return G=(V,E)

Figure 19. RRT algorithm after [16]

An example of a solution obtained with the RRT algorithm is shown in Figure 20. The planner initial location is the red circle on the left, the goal location is the red circle on the right. The blue polygon is an obstacle in the workspace. The RRT algorithm found a feasible path (red), using 59 vertices.

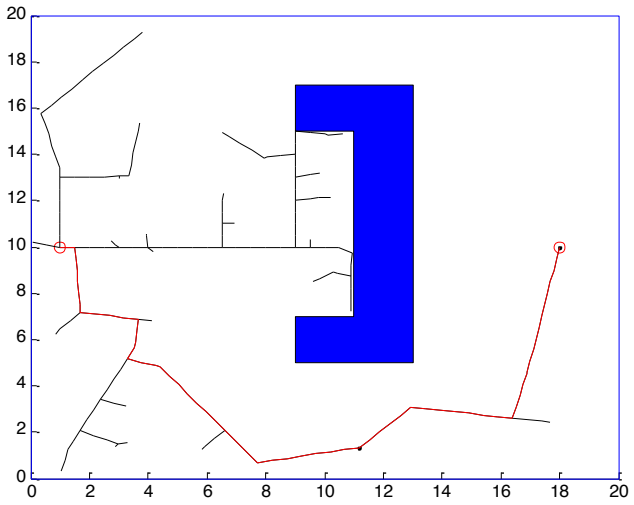

Figure 20. Example of RRT path from initial location on left to goal location on right

As can be seen in Figure 20, the RRT algorithm is a feasible planner and does not yield an optimal path. A variant of RRT, called RRT*, is a near-optimal planner. The algorithm, adapted from [16], constructs and grows the tree in the same way as RRT. However, the tree is augmented with edge costs that capture the path length from the initial point to each vertex. After a new point is added to the tree, the *Near* function finds the set of vertices within a specified radius of x_{near} (i.e., close to the newly added vertex), X_{near}. The *CollisionFree* function then determines if the path between x_{rand} and X_{near} is collision free. If the path is collision free and a path between x_{rand} and the set of vertices in X_{near} is lower cost than the path between x_{rand} and $X_{nearest,}$ then X_{near} is added to V and the path between X_{near} and X_{rand} is added to E (see Figure 21 for the pseudo code). This results in the type of starburst pattern shown in Figure 22.

1: $V \leftarrow \{x_{init}\}$; $E \leftarrow 0$

2: for i = 1,...,n do

3: $x_{rand} \leftarrow \text{SampleFree}_i$

4: $x_{nearest} \leftarrow \text{Nearest}(G=(V,E), x_{rand})$

5: $x_{new} \leftarrow \text{Steer}(x_{nearest}, x_{rand})$;

6: if $\text{ObstacleFree}(x_{nearest}, x_{new})$ then

7: $X_{near} \leftarrow \text{Near}(G=(V,E), x_{new}, \min\{\gamma_{RRT^*}(\log(card(V))/card(V))^{1/d}, \eta\})$

8: $V \leftarrow V \cup \{x_{new}\}$

9: $x_{min} \leftarrow x_{nearest}; c_{min} \leftarrow Cost(x_{nearest}) + c(Line(x_{nearest}, x_{new}))$

10: foreach $x_{near} \in X_{near}$ do // Connect along a minimim-cost path

11: if $\text{CollisionFree}(x_{new}, x_{near}) \wedge Cost(x_{new}) + c(Line(x_{new}, x_{near})) < Cost(x_{near})$

12: $x_{min} \leftarrow x_{near}$; $c_{min} \leftarrow Cost(x_{near}) + c(Line(x_{near}, x_{new}))$

13: $E \leftarrow E \cup \{(x_{min}, x_{new})\}$

14: foreach $x_{near} \in X_{near}$ do //rewire the tree

15: if $\text{CollisionFree}(x_{new}, x_{near}) \wedge Cost(x_{new}) + c(Line(x_{new}, x_{near})) < Cost(x_{near})$

16: then $x_{parent} \leftarrow Parent(x_{near})$

17: $E \leftarrow (E \backslash \{(x_{parent}, x_{near})\}) \cup \{(x_{new}, x_{near})\}$

18: return G=(V,E)

Figure 21. RRT* algorithm after [16]

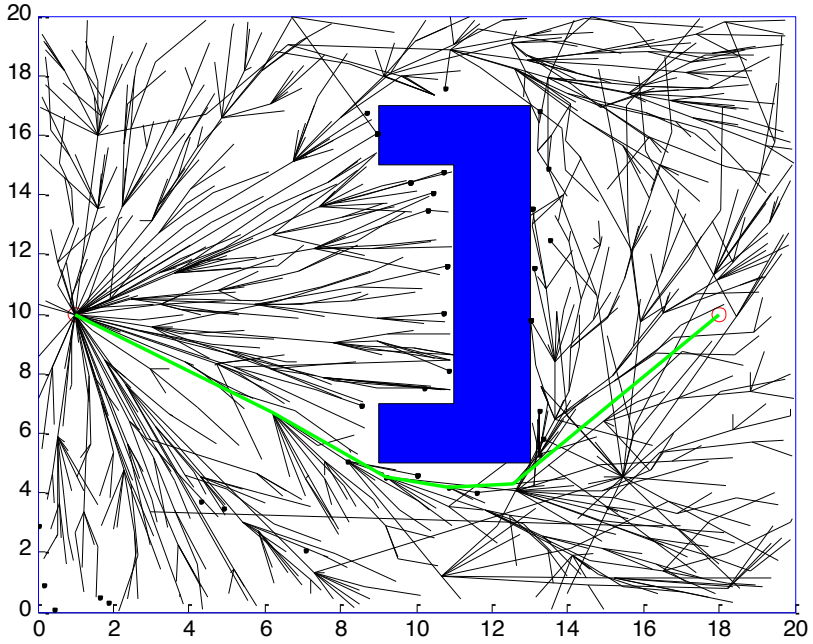

Figure 22. Example of RRT* path from initial location on left to goal location on right

The MATLAB implementation of RRT* uses 1,000 vertices with γ equal to 30 and η equal to 10 and takes about 6 seconds to run. A C implementation of the same algorithm takes less than a tenth of a second for 5,000 vertices! Thus, this planning approach is feasible for real-time implementation and re-planning. The MATLAB implementation is not as efficient for searching the closest neighbor as the tree becomes larger and larger. The current time scale for the MATLAB implementation works in a static environment, but a dynamic environment requires the speed of the C implementation to re-plan paths in real time.

2. Temporal (Velocity) Planning

Once the spatial path is generated, the velocity along the path must be planned while accounting for dynamic obstacle motion (most notably the divers). Since the development platform has independent control of the surge velocity, the proposed approach is to regulate this velocity based on the distance between the diver and the vehicle. The desired following distance for this scenario is one meter. The error between the desired and actual following distances is used with a proportional controller to keep the error to one meter. For safety, the THAUS will only travel forward to increase the distance from the diver, never towards the diver. When the diver swims towards the THAUS, the separation distance will fall below the desired threshold, causing a positive surge control signal to the THAUS that will speed the vehicle up. If the diver slows down, the following distance will increase, causing the THAUS to slow down and eventually stop to wait for the diver to catch up (see Equation 17).

$$\begin{aligned} &\text{control signal} = 1 \text{ - } d_{THAUS,Diver} \quad d \le 1 \\ &\text{control signal} = 0 \quad d > 1 \end{aligned} \tag{16}$$

3. Spatial Path Following

A path-following controller, based on waypoint navigation and cross-track errors, as developed in [22] and [23], is adopted for path following (see Figure 23). The set of vertices that connected the goal to the initial location from the RRT* algorithm define individual straight-line segments and thus define a set of waypoints. The THAUS will

move along the path, pointing towards the next waypoint on the path. The PID controller for yaw developed in Section IV.A is used to point the vehicle at the next waypoint (i.e., heading control). Once within a watch circle radius (0.1 meters in this scenario), the THAUS will select and point to the next waypoint. To minimize cross track error (i.e., the lateral offset of the vehicle from the track), two separate controllers are used, based on the magnitude of cross track error: a Cross Track Error (CTE) controller that controls sway to minimize cross track error, and a Line of Sight (LOS) controller that points the vehicle along the line of sight (perpendicularly) to the track line and drives forward to intercept the track line.

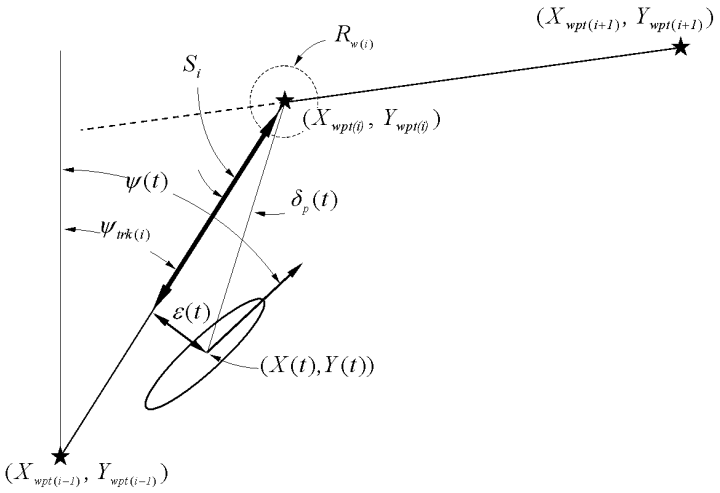

Figure 23. Diagram of waypoint following with cross track error from [23]

Mode 1: A CTE controller is used to keep the THAUS on the track line via the sway channel when the cross track error is less than a specified threshold (one meter for this scenario). The cross track error is calculated by first calculating the distance between the two waypoints defining the current path segment:

$$L_i = \sqrt{(X_{wpt(i)} - X_{wpt(i-1)})^2 + (Y_{wpt(i)} - Y_{wpt(i-1)})^2} \tag{17}$$

The track angle is calculated by:

$$\psi_{trk(i)} = \tan^{-1} \frac{(Y_{wpt(i)} - Y_{wpt(i-1)})}{(X_{wpt(i)} - X_{wpt(i-1)})} \tag{18}$$

The difference between the current position and next waypoint is:

$$\begin{aligned} X_{wpt(i)}(t) &= X_{wpt(i)} - X(t) \\ Y_{wpt(i)}(t) &= Y_{wpt(i)} - Y(t) \end{aligned} \tag{19}$$

The distance from the vehicle to the i^{th} waypoint, projected onto the track line $S(t)_i$, is:

$$S(t)_i = X_{wpt(i)}(t)(X_{wpt(i)} - X_{wpt(i-1)}) + Y_{wpt(i)}(t)(Y_{wpt(i)} - Y_{wpt(i-1)}) \tag{20}$$

The angle between the track line and the line of sight (angle between current heading and heading to next waypoint) is defined by:

$$\delta_p = \tan^{-1} \frac{Y_{wpt(i)} - Y_{wpt(i-1)}}{X_{wpt(i)} - X_{wpt(i-1)}} - \tan^{-1} \frac{Y_{wpt(i)}(t)}{X_{wpt(i)}(t)} \tag{21}$$

Note that the proper quadrant must be verified when using the arctangent function. Finally the cross track error is given as:

$$\varepsilon(t) = S_i(t) \sin(\delta_p(t)) \tag{22}$$

Since the sway degree of freedom of the vehicle can be independently controlled, any cross-track error is driven to zero using the PID controller developed for station keeping in this direction.

Mode 2: the LOS controller is used if the vehicle has greater than 1 meter of cross track error. In this case, the THAUS will point and move towards the closest point on the track until it is 0.5 meters away, before resuming waypoint navigation and CTE control.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS

For this research, the THAUS is to operate in three separate situations:

- station-keeping, where THAUS will move towards and maintain a specified pose without regard to a diver;
- diver-following, where THAUS tracks and follows a diver;
- diver-leading, where THAUS will lead a diver to a goal location.

The technical approaches developed in the previous chapter are applied and evaluated for these scenarios here.

A. STATION KEEPING

The goal of station-keeping is for the THAUS to move into and maintain a specified pose while minimizing offset errors and rejecting disturbances. Two methods are examined to achieve this goal: a PID controller and a potential field method.

1. PID Control Applied to Station Keeping

The PID values from Table 3 in Chapter IV are used both in simulation and experimentation with the THAUS in the CAVR test tank. The results in Figures 25-27 show the response of the simulation and the THAUS to several changes in pose. These changes are performed simultaneously for the three channels (i.e., coupled motion) to evaluate the response of the vehicle to go from an initial to a final pose. Recall that it is assumed that no coupling occurs between the channels for the development of the simplified dynamic model. In reality the THAUS uses vectored thrusters: the same four thrusters control surge, sway, and yaw motion. Some coupling between the different channels is expected due to the vectored thrusters. However, the THAUS is able to respond to the changes in pose and achieve minimal stead state error (based on the length of time) even with this restrictive assumption. From the figures it can be seen that the controllers developed in simulation perform adequately on the actual vehicle as well. The slight difference in model vs. vehicle response is not important here.

PID controllers have the potential to effectively reject disturbances. The integral controller in particular can achieve a zero steady-state error even with disturbances like

currents. Due to the limitations of the experimental setup, disturbance rejection was not tested, but will be evaluated in future work.

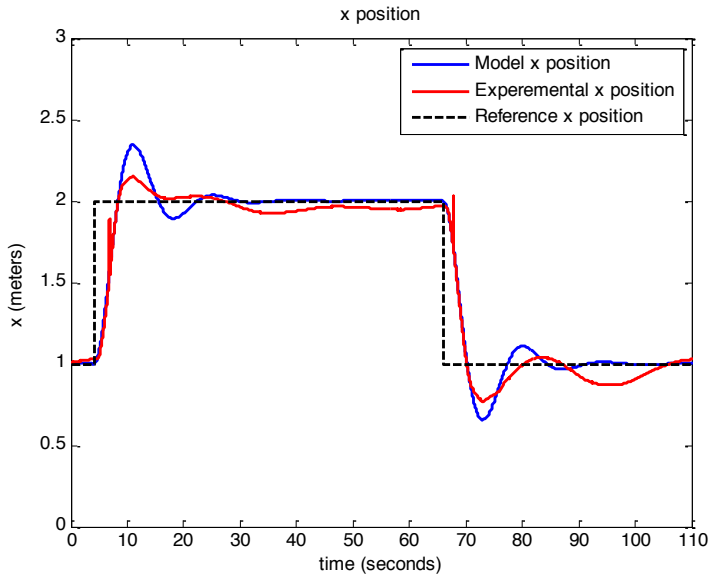

Figure 24. Step responses in surge with PID position control and coupled motion.

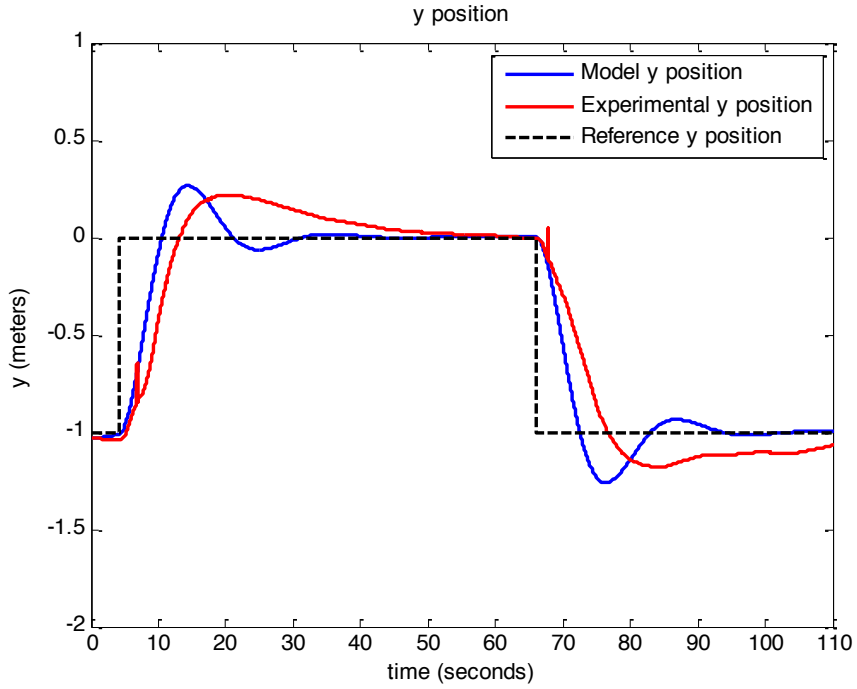

Figure 25. Step responses in sway with PID position control and coupled motion.

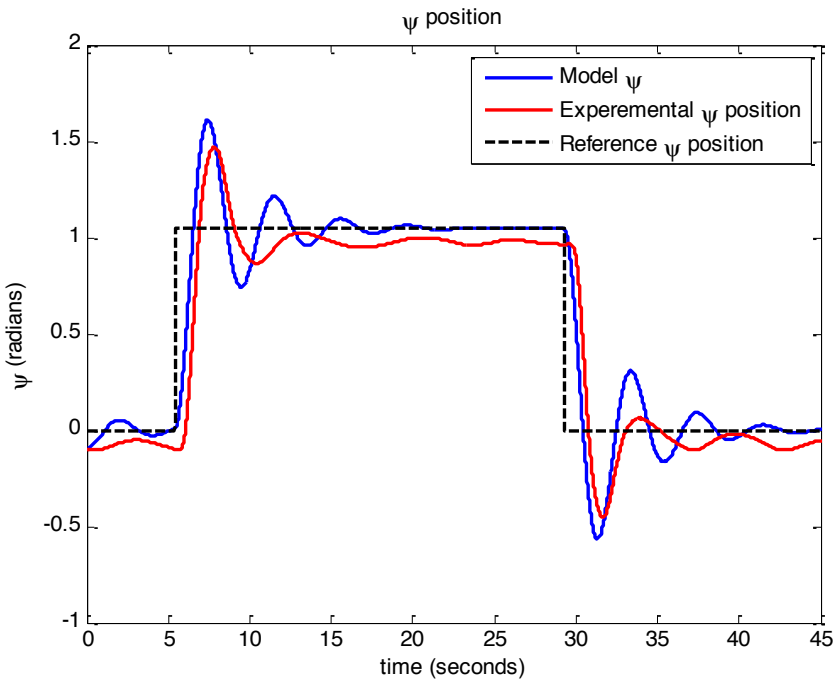

Figure 26. Step responses in yaw with PID position control and coupled motion.

2. Potential Field Method Applied to Station Keeping

To illustrate the potential field method for station keeping, Figures 28-31 show the response of the vehicle using a potential field superimposed with results from the PID controller (both in simulation). The potential field method has a slower response than the PID controller due to the lack of derivative control in the potential field approach. Although the potential field method shows zero steady-state error, the THAUS has a dead band that is not modeled where the integral control is necessary to cancel out any steady state error.

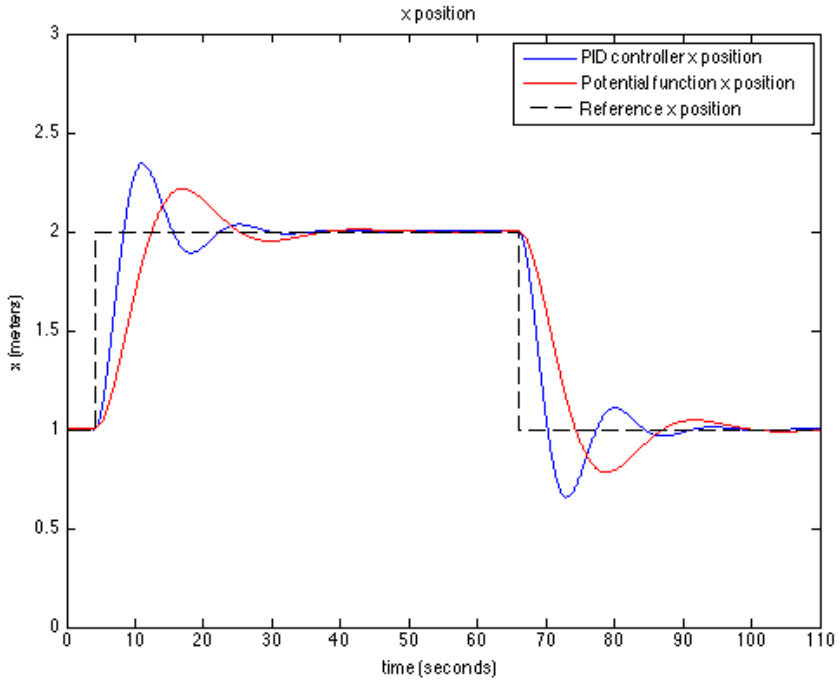

Figure 27. PID and PF response in x direction

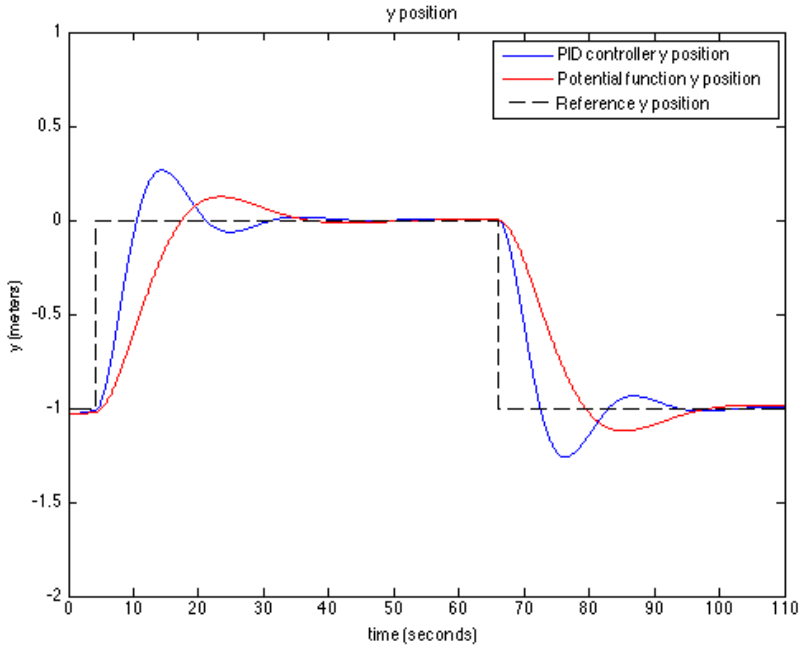

Figure 28. PID and PF response in y direction

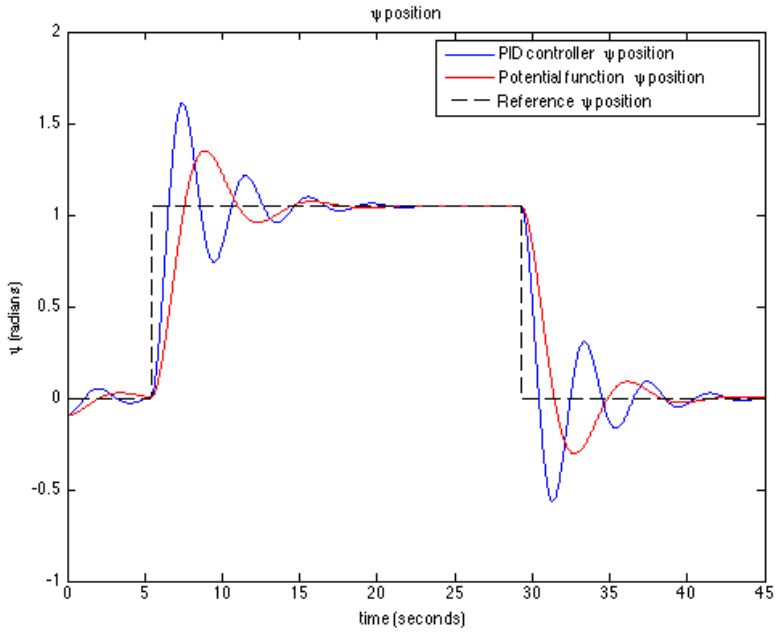

Figure 29. PID and PF response in ψ direction

3. Discussion

The PID controller and potential function are both viable solutions for station-keeping operations. The potential field method is less tunable and has a slower response than the PID controller due to the lack of integral and derivative controllers. Overall the advantages in tuning the PID controller and disturbance rejection make the PID controller better for station-keeping operations.

B. DIVER FOLLOWING

For diver-following, the THAUS must react to and follow the diver while ensuring diver safety. The results of using a PID controller and potential field method are presented. Deliberative approaches are unable to currently predict diver behavior and are not used for diver following. This will be addressed in future works.

1. PID Control Applied to Diver Following.

To demonstrate the PID control scheme applied to diver following, Figure 30 illustrates the results when the station keeping PID controller is used to move the THAUS

from an initial pose to the goal location. The goal pose (green arrow) is on the opposite (right) side of the virtual diver (black arrow) from the initial pose (red circle). Since the method does not account for obstacles, the THAUS path (blue) passes through the diver when moving to the goal. This is not unexpected since the PID control approach does not contain a mechanism for obstacle avoidance. This method is inadequate to ensure diver safety.

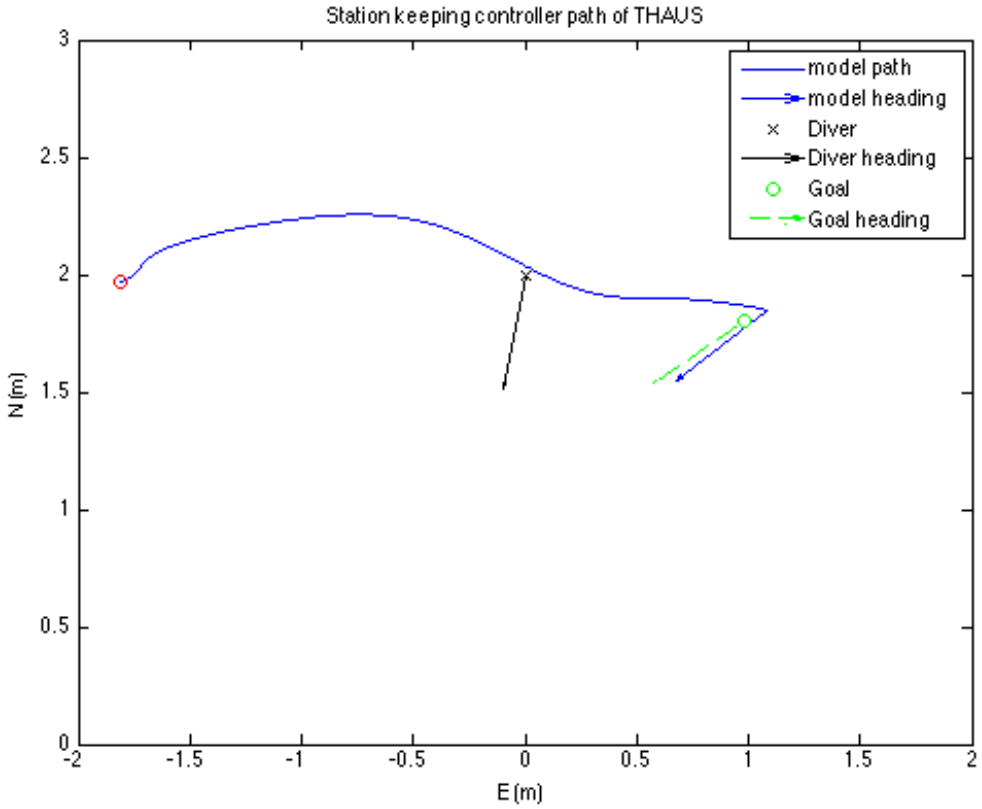

Figure 30. THAUS path using PID control only

2. Potential Field Method Applied to Diver Following

To illustrate the application of the Potential Field method to diver following, Figure 31 shows a simulated run (blue) in the same scenario as Section IV.B.1. The THAUS starts at the red circle on the left side of the diver (black) with the goal (green) on the right side of the diver. The repulsive potential keeps the THAUS away from the

diver and the THAUS is able to navigate around the diver to the goal. On the same figure, the measured path of THAUS (red) navigating the same scenario in the CAVR test tank and using the reactive scheme verifies the desired behavior. Headings are indicated with arrows.

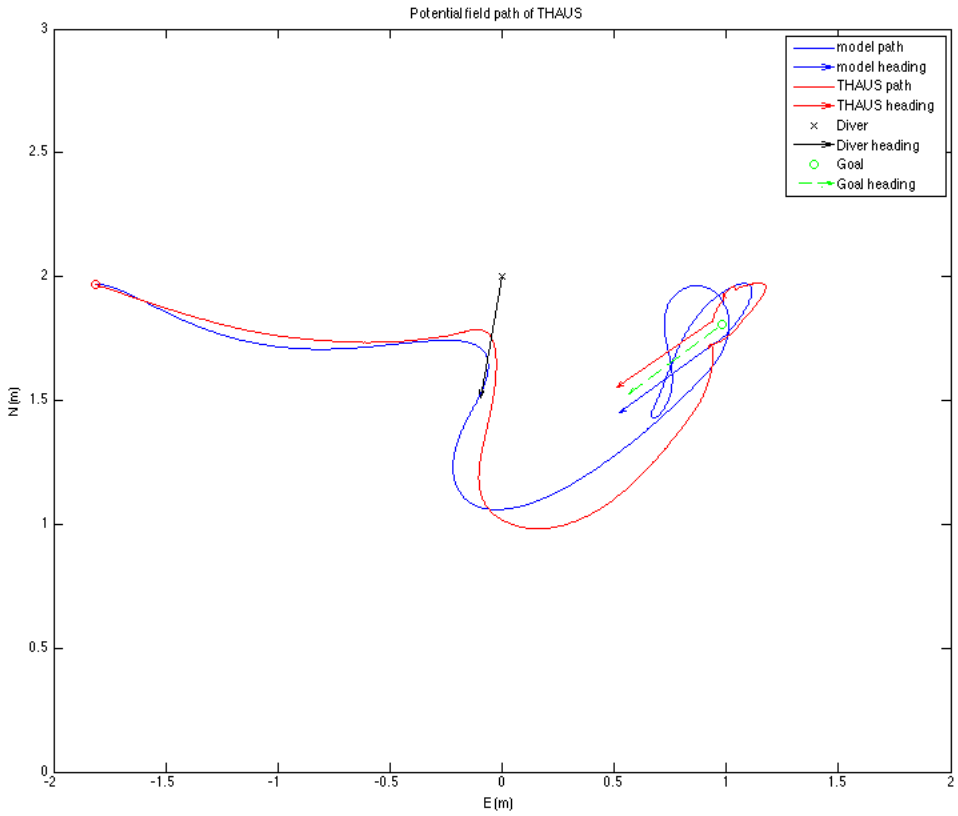

Figure 31. Simulated (blue) and measured (red) paths using the potential field method.

Figures 32 and 33 show the attractive and repulsive gradients, respectively acting on the vehicle as a function of time. These functions do not line up exactly since the potential is a function of the vehicle position, which differs slightly for the simulated vs real vehicle. This offset is not a concern here since the objective is to demonstrate that the method developed based on the simulated system responds appropriately on the vehicle as well.

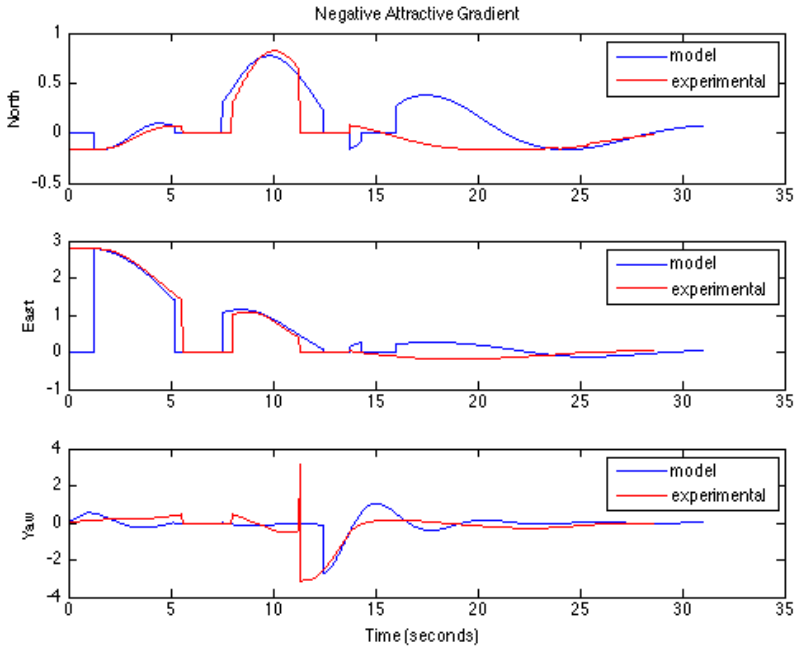

Figure 32. Attractive gradient results in NED

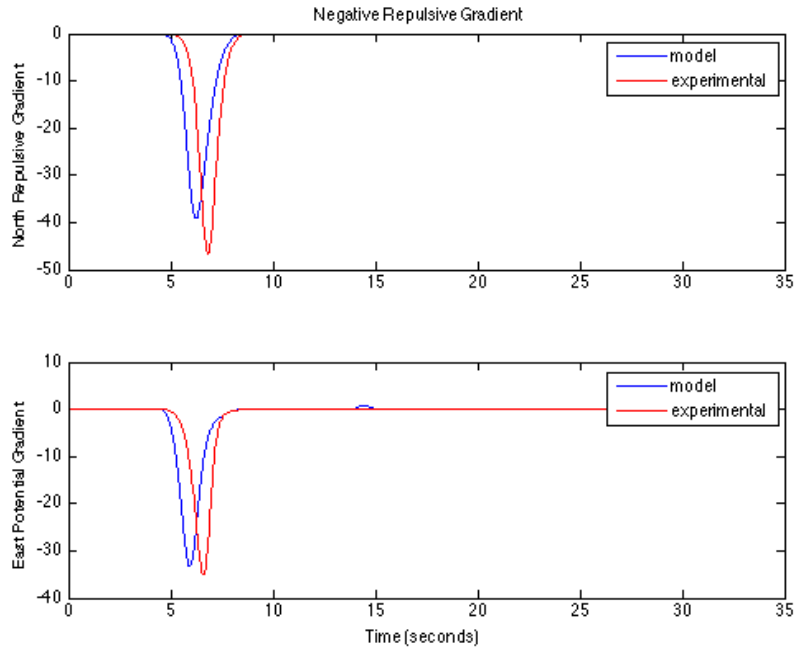

Figure 33. Negative gradient results in north and east directions

These results are applicable to a stationary diver (e.g., a diver performing a task in the workspace while the robot provides utility such as illuminating the workspace). For the next experiment, the virtual diver moved forward at a velocity of 0.4 m/s. Figure 34 shows the absolute error in meters between the THAUS and diver-relative goal pose.

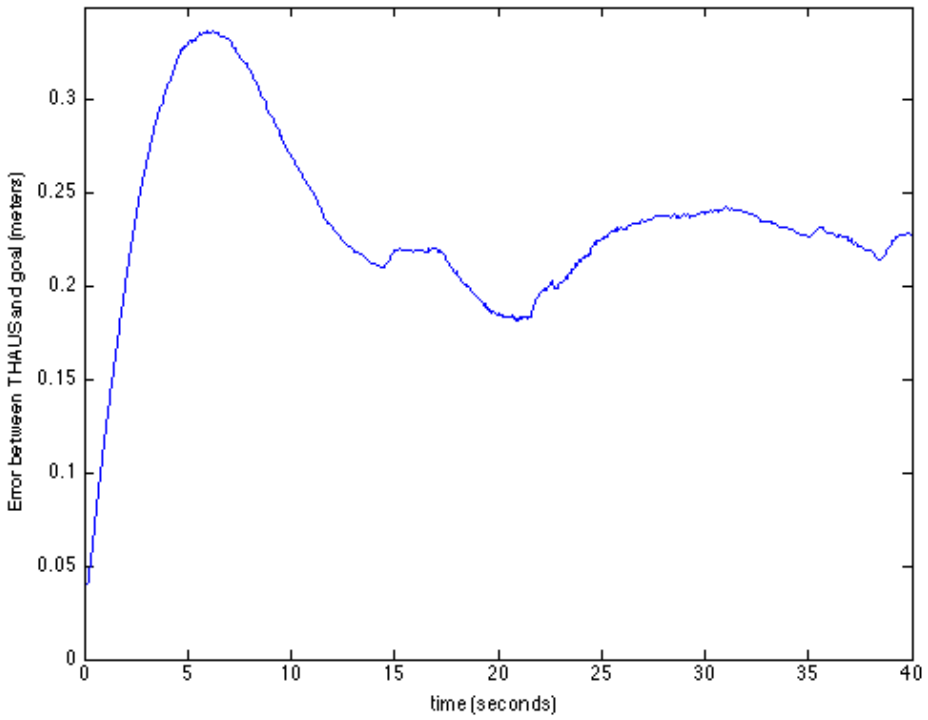

Figure 34. Distance error between goal location and THAUS

The THAUS has an initial lag behind the diver since it accelerates from rest (whereas the diver is moving). Due to the constant velocity of the diver and the proportional potential function of the THAUS, the THAUS lags behind the diver by 0.2 meters. Extensions to the potential field function are introduced in the next section to improve the system performance when following a moving diver.

3. Potential Field Method Performance Improvement

A disadvantage to using the potential field as defined before is that it is reactive to the diver's instantaneous position only (i.e., it takes the current most appropriate action

without regarding past or future actions or consequences). As a result, the THAUS will always lag behind the goal if the diver is moving. Two improvements are proposed to the attractive potential to speed up the response and minimize steady state error. To speed up the response to the diver's motion, an additional attractive potential function is added to match the vehicle's velocity with diver's velocity:

$$\nabla U_{vel}(V) = \zeta_v (V - V_{goal}) \tag{22}$$

V is the velocity of the THAUS and V_{goal} is the velocity of the goal (i.e., the diver). This term is tuned to provide a rapid input to the vehicle as soon as the diver moves. This is a similar structure to a derivative controller in Section IV.A that drives the rate of change of the error to zero. Another potential gradient

$$\nabla U_{\text{int}} = \zeta_{\text{int}} \int (q - q_{goal}) dt \tag{23}$$

is added to drive the accumulated steady state error in position to zero. This potential will increase over time if the error between the THAUS and goal persists (in the same way that an integral controller in Section IV.A works). This potential acts to overcome nonlinearities such as deadband in the vehicle response at low excitation levels.

Figure 35 shows a comparison between the two methods. The Figure shows that the augmented approach has a quicker response, less peak error, and smaller steady state error than the standard potential field alone (the error never reaches zero due to the size of the dive tank).

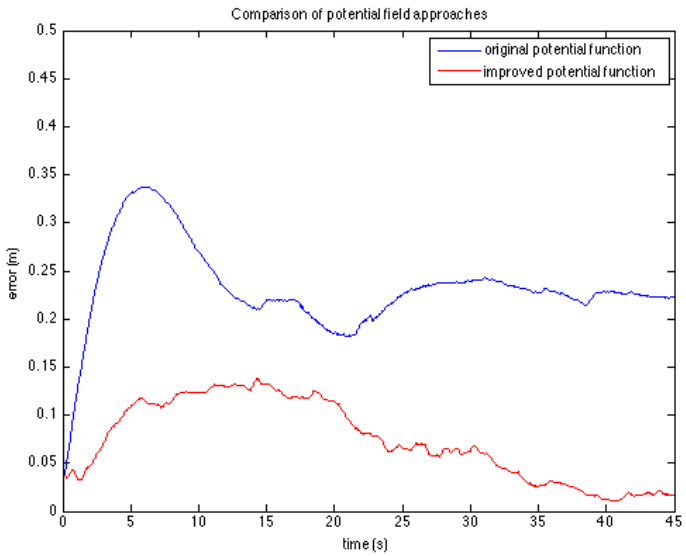

Figure 35. Experimental comparison of the two potential field approaches

4. Discussion

The PID controller is not a viable solution for diver-following operations due to the lack of obstacle avoidance. The potential field method is a viable solution to have the THAUS navigate around the diver into a goal position. Adding the integral and derivative potentials to the attractive potential only when the diver is moving improved transient and steady state response. Deliberative approaches are unable to currently predict diver behavior and are not used for diver following.

C. DIVER LEADING

For diver-leading operations, the THAUS must lead a diver through an obstacle field to a goal location. The results for the potential field method and deliberative approach are presented. The PID control is not applicable here as there is no obstacle avoidance mechanism (as demonstrated Section V.B.1).

1. Deliberative Planning Applied to Diver Leading

For this scenario, the THAUS is to lead the diver from an initial location, around an obstacle, to a goal location. Figure 36 shows the scenario with the initial position on the left

(red circle), obstacle (blue polygon), and goal location on the right (red circle). The path (green) generated from RRT* connects the initial to the goal location around the obstacle.

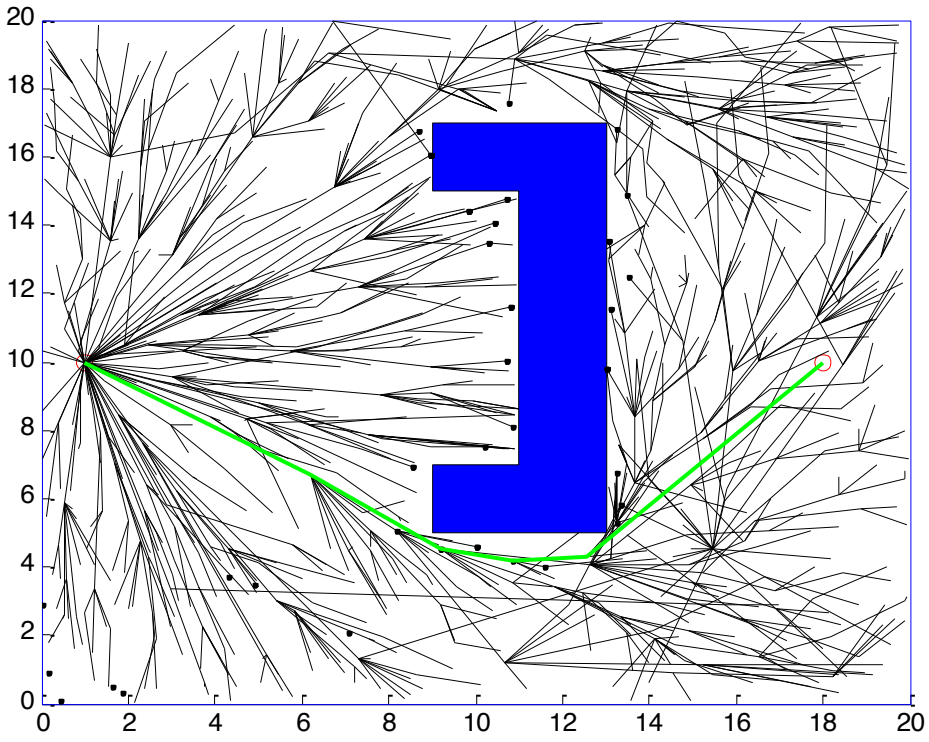

Figure 36. Diver leading scenario from initial location on left around the obstacle (blue) to goal location on right.

The RRT* algorithm generates a path for the THAUS to follow using the temporal and spatial following controllers. The RRT* generation used 1,000 nodes with γ equal to 30 and η equal to 10. Figure 37 shows a simulated run with the obstacle in several time steps. The upper left plot shows the THAUS in red, offset from the path in green. The path moves around the blue obstacle. The diver, in black, is 3.5 meters away from the THAUS. The upper right plot shows the diver swimming towards THAUS while THAUS uses LOS control to point directly at the path. As the diver swims towards the THAUS, the THAUS begins moving towards the path. Once within 0.5 meters of the path the THAUS turns back onto the path (left, center) and continues waypoint

navigation. The speed of the THAUS continually adjusts to maintain a one meter offset from the diver (middle two plots). The THAUS uses the cross track error controller for the remainder of the run to track the path and the LOS controller to make course corrections). The THAUS continues following the path, leading the diver around the obstacle to the goal location (bottom two plots).

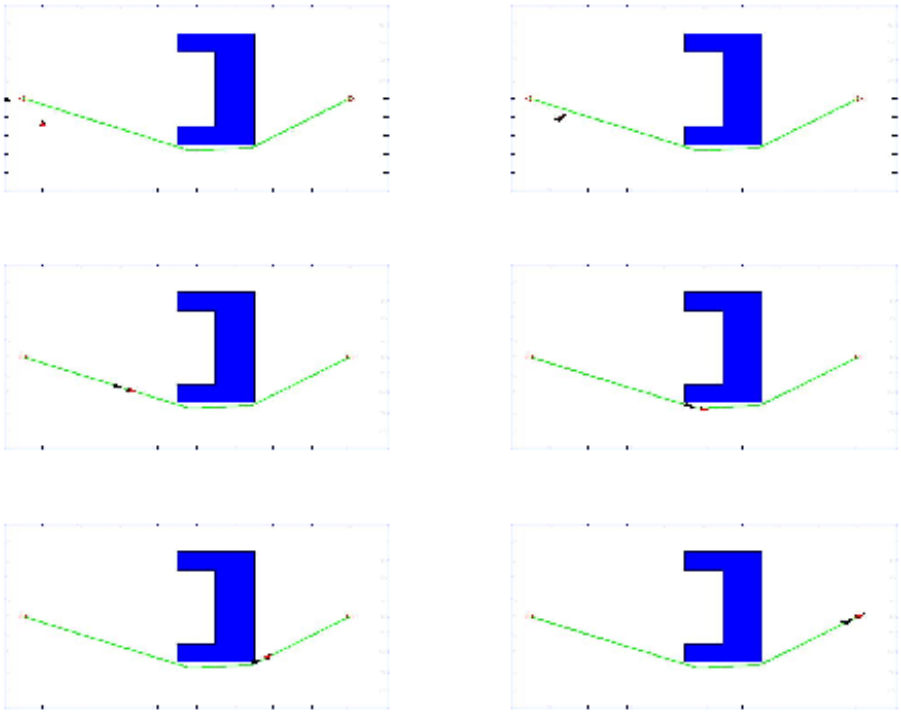

Figure 37. The THAUS leads a diver around an obstacle to a goal location

Figure 38 shows the relative distance between the diver and the THAUS. At steady state, the separation distance remains close to the specified threshold. As the diver initially closes in on the vehicle, the distance between the diver and THAUS decreases to less than 0.5 meters. This is not a safety concern since the diver is controlling the distance to the THAUS and the THAUS is reacting to the diver by moving away. Figure 39 shows the cross track error of the THAUS. The THAUS is offset from the path at the start of the run to illustrate the two modes of operation in path following. The cross track error is

minimal until the THAUS turns towards the next waypoint. The cross track error increases as the momentum from the previous track carries the THAUS off the intended path. The cross track error controller then reduces this error within five seconds to remain on the new track.

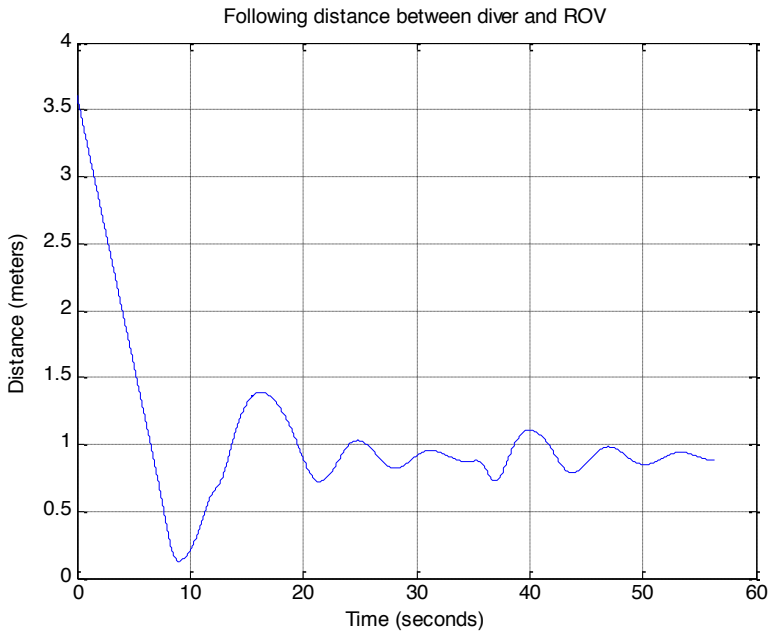

Figure 38. Following distance between diver and THAUS

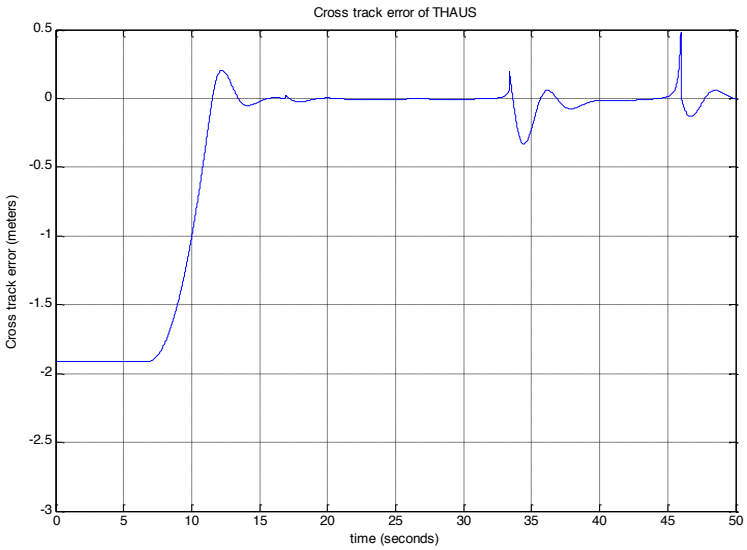

Figure 39. Cross track error of the THAUS

2. Potential Field Approach Applied to Diver Leading

Figure 40 illustrates the potential field method applied to the same scenario. The THAUS starts off on the left and moves directly towards the goal on the right (since it is a reactive method) but is repelled by the obstacle. Due to the shape of the obstacle, the THAUS gets stuck in the local minimum in the potential field and does not reach the goal location. The potential field approach is inadequate to maneuver in a generic environment with obstacles.

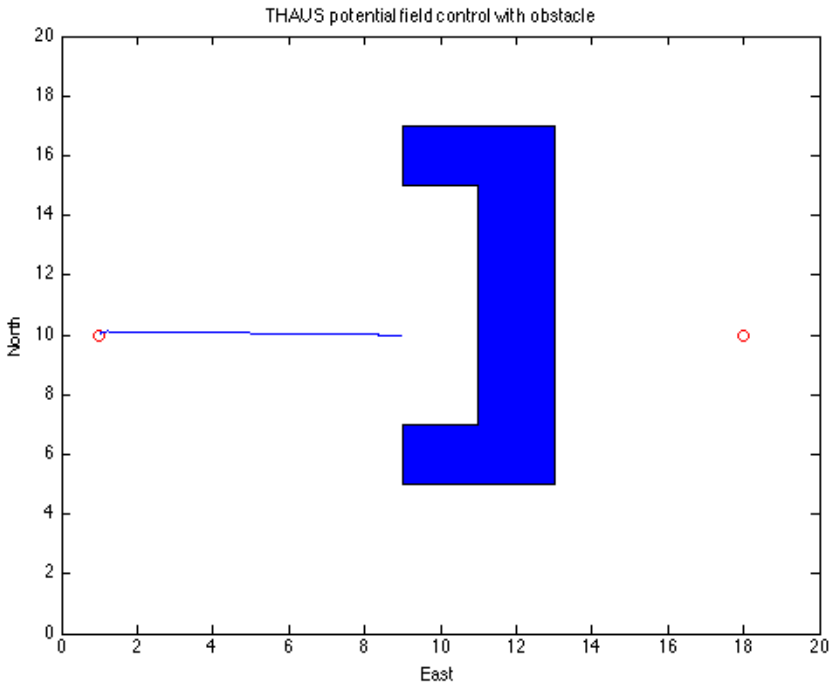

Figure 40. THAUS path using potential field methods.

3. Discussion

The potential field method is inadequate to lead the diver around an obstacle to a goal location due to the potential for the THAUS to get stuck in a local minima. RRT* coupled with spatial and temporal path followers is a viable solution to successfully lead a diver around an obstacle to a goal location.

VI. CONCLUSIONS

A. SUMMARY

Close-quarters operations with AUVs allows for the exploration of novel applications, including joint diver-robot operations. However, diver safety is of paramount importance when the human and robot must share a workspace. The objective in this work is to enable three modes of operations: station-keeping, diver-following, and diver-leading. Reactive and deliberative methods are investigated in this work to facilitate these modes of operation.

A simplified, decoupled dynamic model for a hovering-class AUV is developed for the SeaBotix vLBV300 platform. The degrees of freedom modeled are in the horizontal plane: surge, sway, and yaw. The decoupled models for each degree of freedom is developed using step inputs and measuring the resulting output. This gives three separate frequency-domain transfer functions, which are combined to yield a simplified dynamic model of the vehicle for planar motion. This model is used to develop reactive diver avoidance and robot positioning in simulation before applying these techniques to the real system.

Different technical approaches are developed based on the mode of operation of the THAUS. For station-keeping, a PID controller and potential field method are used to command the vehicle into a specific pose. For diver-following operations, both the PID controller and potential field method are investigated to operate in close quarters to a virtual diver. Finally the potential field method and RRT* path planning algorithm, combined with a cross track error controller, is used to lead a diver around an obstacle towards a goal location.

For station-keeping operations, the PID controller showed better results over the potential field method. The potential field had a slower response time. Additionally, without the integral control developed for the PID controller, the vehicle's dead-band and any environmental disturbances will prevent the vehicle from reaching the desired pose, resulting in a steady-state error.

For diver-following, the potential field method outperformed the PID controller. The PID controller does not avoid the diver and no mechanism exists to keep the diver safe. The PID controller is also susceptible to integral windup, which can cause the vehicle to rapidly approach the goal location. The potential field method showed a smooth trajectory, navigating to the goal location while avoiding the diver, as desired.

For diver-leading, the path planning algorithm outperformed the potential field method. The potential field method is prone to local minima. This problem requires a deliberative approach in complex operating environments. Utilizing the RRT* for path planning and the waypoint navigation coupled with the cross-track error controller, the vehicle responded to the diver to maintain a specified distance, navigated around obstacles, and lead the diver to a goal location. However, application of this approach to diver following is challenging since knowledge of the diver's expected behavior is required.

B. FUTURE WORK

Based on the results achieved in this work, as well as the limitations identified, several areas for future work have been identified. First, a coupled, 6-DOF hydro-dynamic model of the development platform has recently been developed, using a state-space representation. Time domain modeling (as opposed to frequency domain analysis, as used in this work) provides additional insight into the vehicle dynamics, accounting for coupling between channels, and allows for the application of robust control algorithms like LQR. The developed modeled utilizes individual thrusters, enabling an additional degree of freedom: roll. Finally, the approach is adaptive: if the vehicle dynamics change (e.g., THAUS uses a grabber arm to pick up an object), the model and control can adapt to account for this change. All of these results facilitate tighter control of the vehicle during proximal operations (including operations among divers). The results obtained in this research will directly benefit from integration with the new vehicle model.

Second, reliance on the external motion capture arena allowed for rapid generation of initial results, but the size of the test tank proved to be restrictive. One solution is to add onboard sensors (e.g., an inertial navigation system and Doppler

velocity log) to track the vehicle pose. This in turn allows for the obtained techniques to be applied in the real-world, open-ocean environment. The obtained approaches need to be exercised in the controlled dive tank and open-ocean to verify the feasibility of the approach to navigate among diver, in particular in the presence of environmental disturbances.

For the current work, a virtual diver is assumed, but this is obviously restrictive. Ideally, a diver tracking and prediction system, either acoustic or optical, needs to be developed to complete this work. The close-quarters operations with divers allow a visual recognition system to be used that would normally be infeasible in an underwater environment. Depending on ambient lighting and water clarity, an acoustic tracking system may work better. This could either be a transmitter on the diver that the vehicle will detect, or a reflector worn on the diver that the vehicle will ping and receive the data back. Alternatively, an external localization system (e.g., an Ultra-Short Baseline acoustic beaconing system) can also track divers in the workspace, but accuracy is limited and such an approach requires infrastructure deployment.

A true deliberative approach will require an accurate predictive model of diver motions. Such a behavioral model will allow the robot to reason about current and future diver actions are is required to truly integrate the diver assistant into a collaborative diver-robot team.

Finally, both the virtual diver's and THAUS' pose are assumed to be known at all times. There will be uncertainty associated with the robot and diver locations in addition to environmental disturbances. These uncertainties must be explicitly accounted for in planning operations. Additional future work entails modeling diver behavior to predict their motion into the future. This will allow the addition of deliberative strategies in the diver-following mode of operation.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

First Order MATLAB Script

```
% first_order.m
% calculates open loop transfer function of a first order response
based on
% the time constant and steady state value

clear all; close all; clc

% Data from velocity response
load fomodel



x=measure_exp(:,1); % x measurement from Vicon
y=measure_exp(:,2); % y measurement from vicon
time=time-time(1); % resets the initial time to start at zero
x=x-x(1); % initial x starts at 0
y=y-y(1); % initial y starts at 0
pos=sqrt(x.^2+y.^2); % straightline distance from origin

vel=zeros(numel(time),1); % intializes velocity
vel(2:end,1)=diff(pos)./diff(time); % differentiates position by time
for velocity

% data before input was received that is eliminated
a=min(vel);
[a,b]=find(vel==a);
time(1:a)=[];
vel(1:a)=[];
time=time-time(1);
vel=vel-vel(1); % initial velocity starts at zero
vel=smooth(vel); % differentiated velocity is smoothed out

ss=mean(vel(time>7)); % steady state value of velocity
tc_amp=0.63*ss; % time constant amplitude is 63% of steady state value
tc=interp1(vel,time,tc_amp); %t ime constant is corresponding x value
of the amplitude
a=1/tc; % inverse of time constant for transfer function

KpKs=a*ss; % gain value based on steady state amplitude
Ks=500; % input value of joystick
Kp=KpKs/Ks; % gain of plant for transfer function
G=tf(Kp,[1 a]) % transfer function

% plot of smoothed data with step response of transfer function
figure
stepplot(Ks*G)
hold on
plot(time,vel,'r')
ylabel('velocity (m/s)')
```

```
axis([0 time(end) 0 .5])
title('First order comparison of model and experimental results')
legend('model','experimental')
```

Second order MATLAB script

```
% second_order.m
% calculates second order open loop transfer function based on natural
% frequency, peak time, percentage overshoot, and damping ratio

close all; clear all; clc

% Data from position response
load somodel.mat

y=measure_exp(:,2); % sway position from Vicon

% deleting initial data before control signal
[a,b]=find(reference_exp(:,2)==0);
time(a)=[];
y(a)=[];
time=time-time(1); % initializing time to zero
y=y-y(1); % setting origin at start



cfinal=mean(y(time>10)); % steady state value
percentos=(max(y)-cfinal)*100; % percent overshoot
zeta=-log(percentos/100)/sqrt(pi^2+log(percentos/100)^2); % damping
ratio
Tp=time(y==max(y)); % peak time
wn=pi/(Tp*sqrt(1-zeta^2)); % natural frequency
Ge=tf([wn^2],[1 2*zeta*wn wn^2]); % closed loop plant
G=tf([wn^2],[1 2*zeta*wn 0]) % open loop plant

% comparison of position responses
figure
step(Ge)
hold on
plot(time,y,'r')
axis([0 time(end) 0 1.5])
ylabel('Position (meters)')
legend('model','experimental')
title('Comparison of second order responses')
```

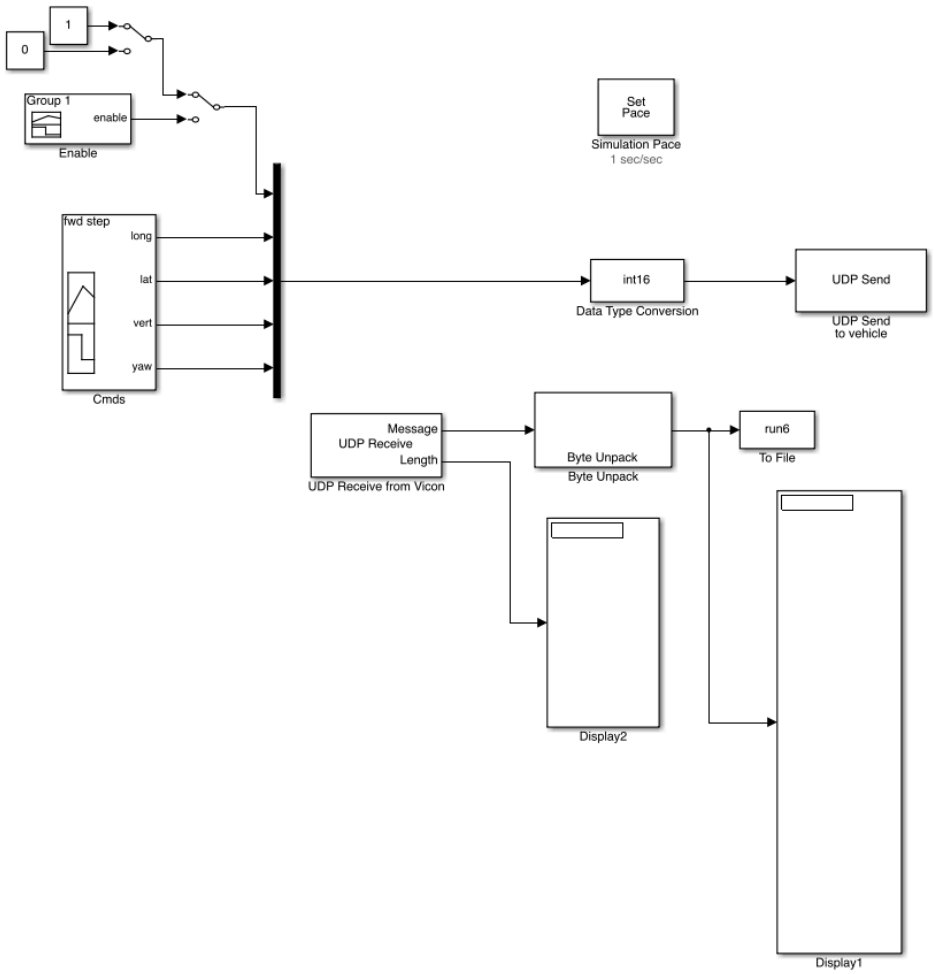

Figure 41. Simulink diagram of experimental setup to deliver commands to THAUS and receive data from Vicon.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B

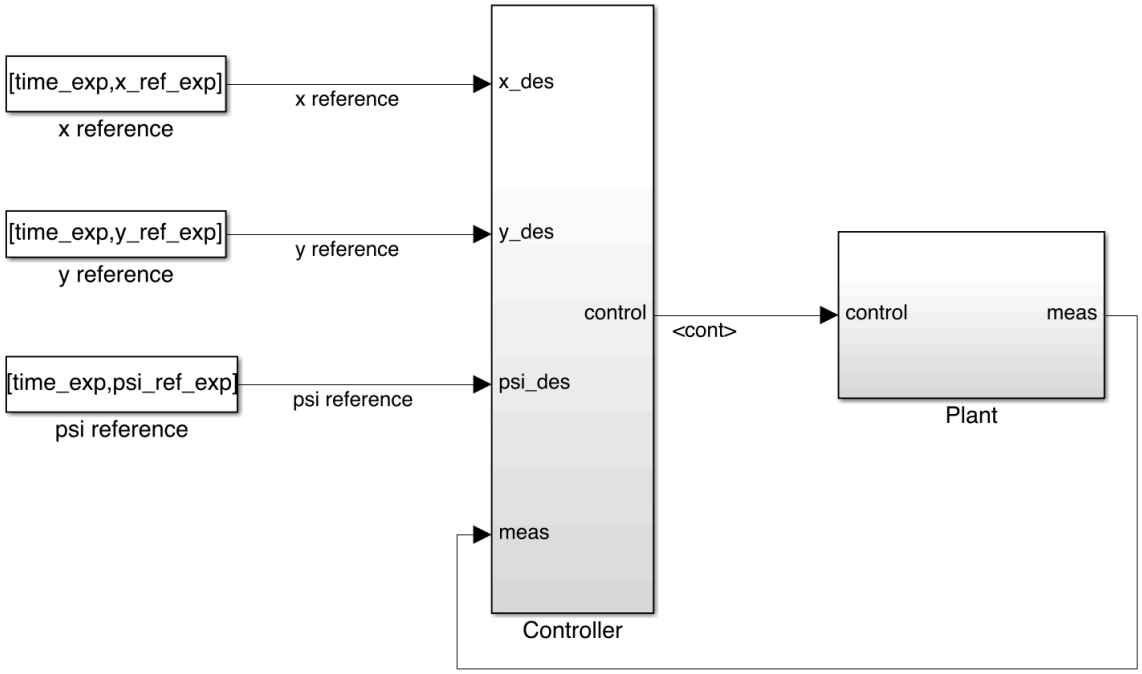

Figure 42. Simulink block diagram of model plant and controller.

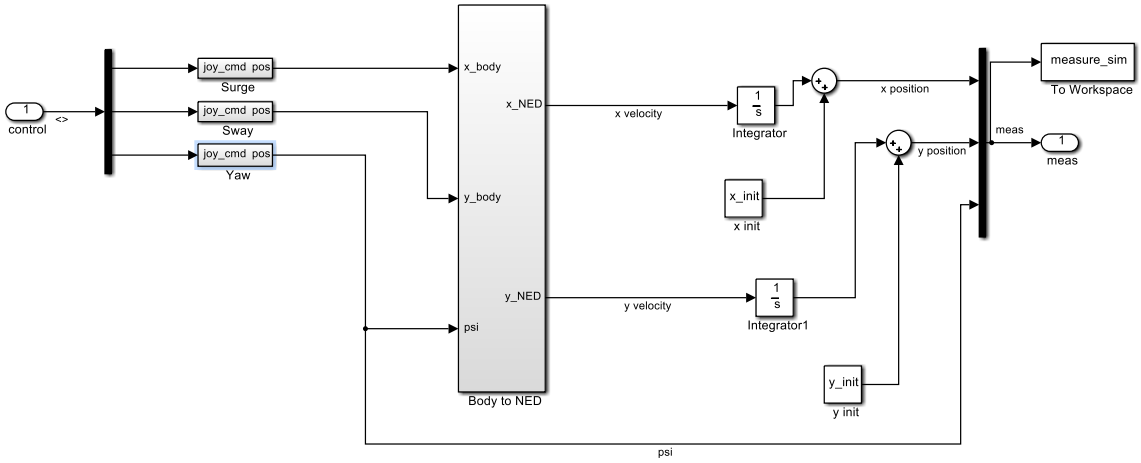

Figure 43. Simulink block diagram of THAUS

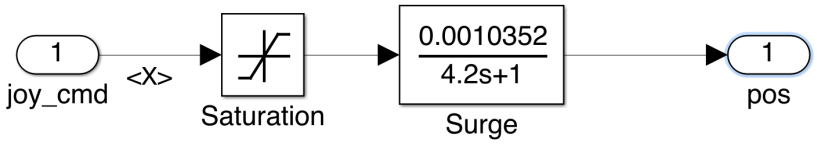

Figure 44. Simulink block diagram of THAUS surge channel

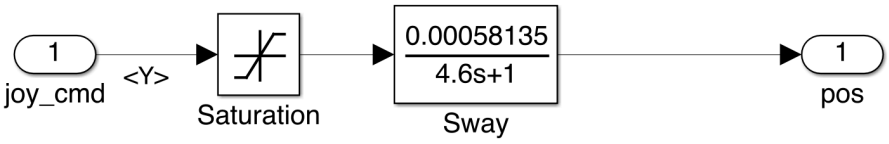

Figure 45. Simulink block diagram of THAUS sway channel

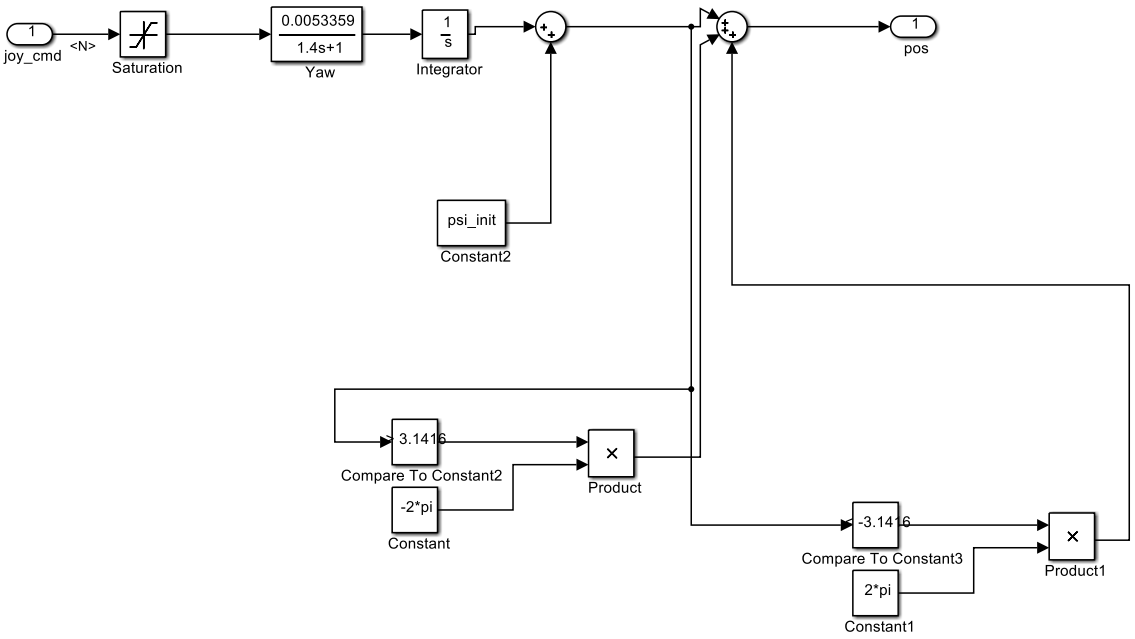

Figure 46. Simulink block diagram of THAUS yaw channel. The block includes logic to have output limited to + or – π.

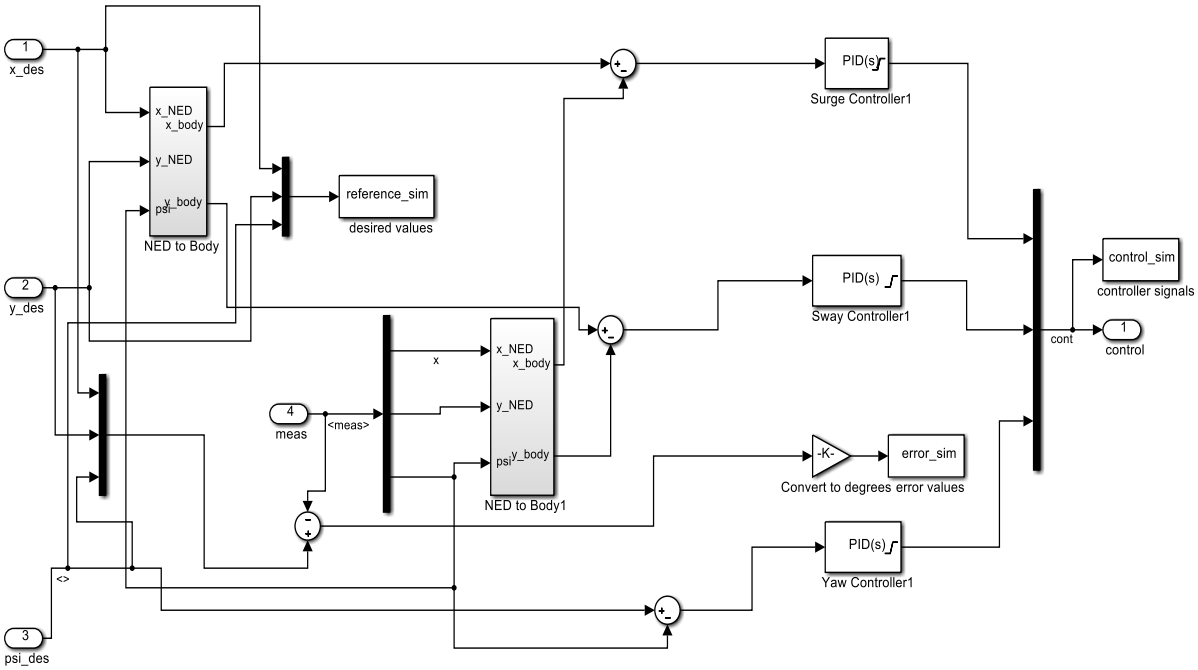

Figure 47. Simulink block diagram of PID controller. The reference and measured signals are both rotated to the body frame before being subtracted for the error signal. Each channel is then fed into its own PID controller.

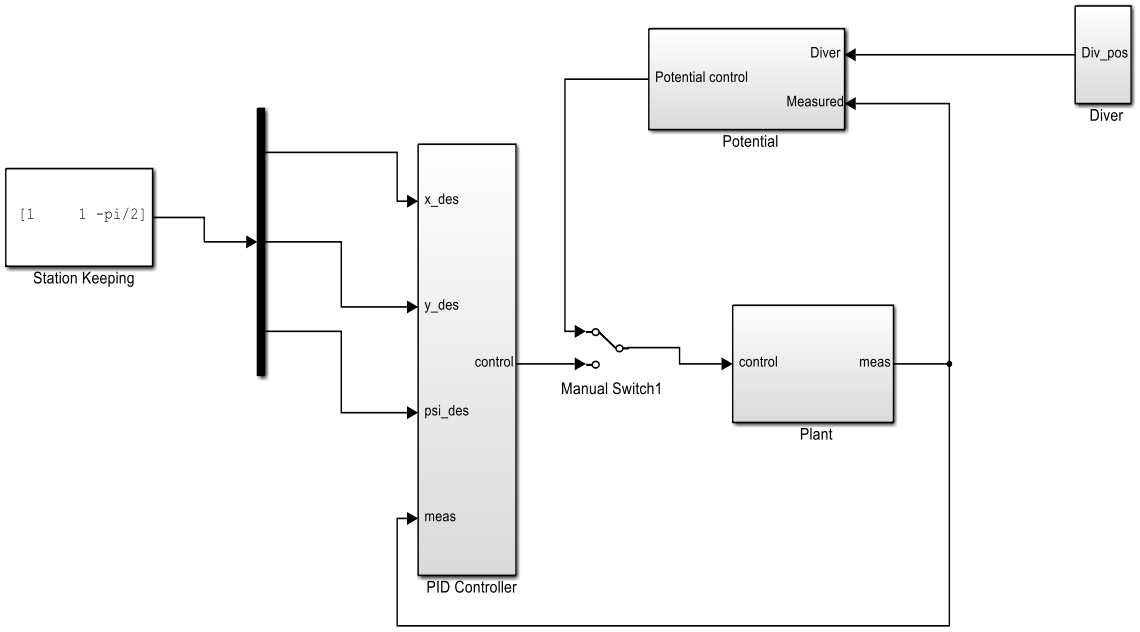

Figure 48. Simulink block diagram of potential controller integrated with station keeping

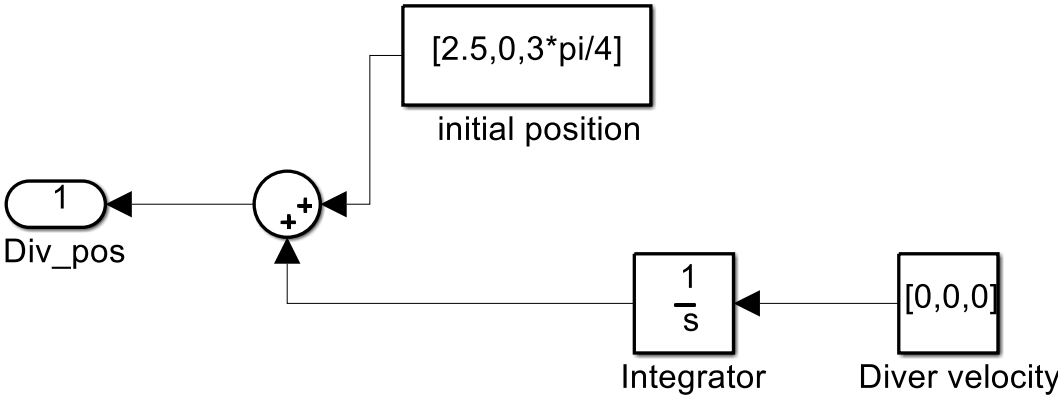

Figure 49. Simulink block diagram of diver motion

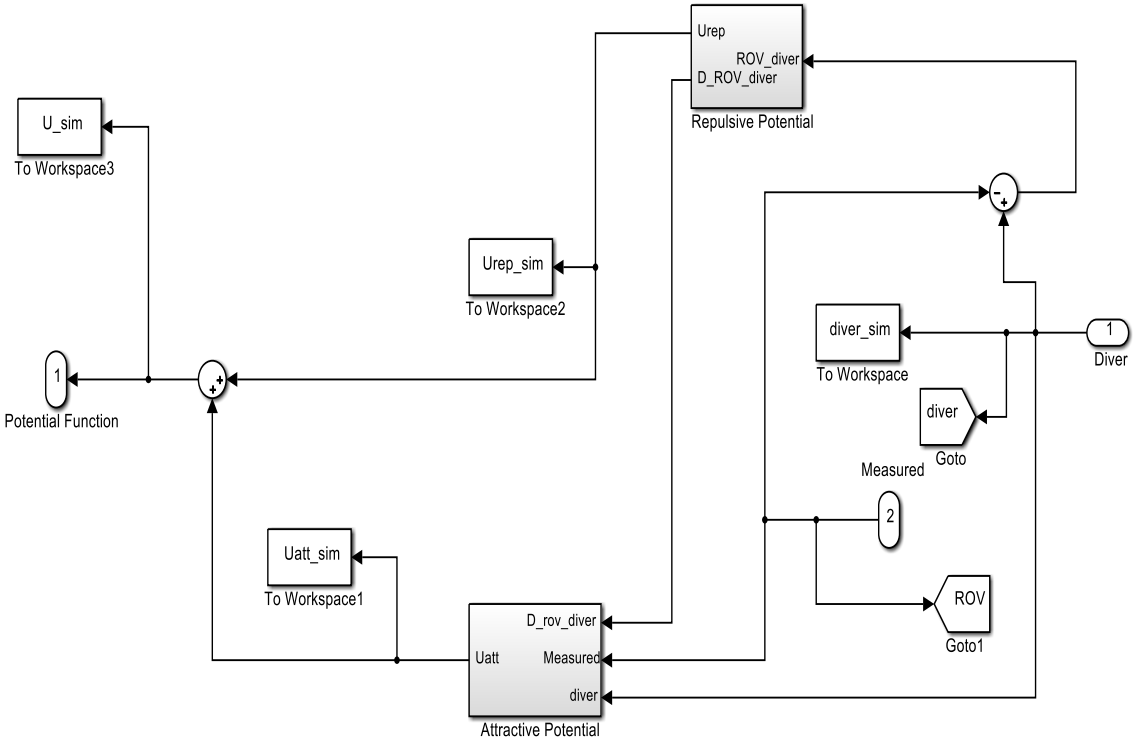

Figure 50. Total potential is made up of repulsive and attractive potential blocks.

RRT MATLAB script

```
%RRT.m
% RRT algorithm with one obstacle
clc; close all; clear;

% vehicle start position
x_init=10;
y_init=1;

step_init=.5; % step size
step_size=step_init;
n=1000; % number of attempts to expand the tree
edges=[];
path=[];
figure()
hold on
```

```
% workspace
xworkspace=[0;0;20;20;0];
yworkspace=[0;20;20;0;0];
plot(xworkspace,yworkspace)


x_obs=[0,0,1,1,0,0,2,2,0]*2+9;
y_obs=[0,1,1,5,5,6,6,0,0]*2+5;

fill(x_obs,y_obs,'b')


q_init=[y_init,x_init]; % tree root is at vehicle position
plot(q_init(1),q_init(2),'ro')
q_goal=[18,10]; % q goal position on opposite side of obstacle
plot(q_goal(1),q_goal(2),'ro')

%% Build RRT
Tree_init=[q_init];
Edge_init=[];
Tree_goal=[q_goal];
Edge_goal=[];
for i=1:n
    collision=1;

    while collision~=0 % selects a random point not in the obstacle
        q_rand=[randi(20*n/10)/(n/10),randi(20*n/10)/(n/10)];
        collision=sum(inpolygon(q_rand(:,1),q_rand(:,2),x_obs,y_obs));
    end

    % Extend RRT

    [IDX,D]=knnsearch(Tree_init(:,1:2),q_rand); % finds closest point
on tree
    q_near=Tree_init(IDX,1:2);
    angle=atan2d([q_rand(2)-q_near(2)],[q_rand(1)-q_near(1)]); % grows
branch out towards q_rand
    q_new=q_near;

    while sum(q_new~=q_rand)~=0 % grow branch until it reaches q_rand
       q_old=q_new;
       d=pdist([q_new;q_rand]);
        if step_size>d
            q_new=q_rand;% remaining distance to goal
        else
        q_new=q_new+step_size*[cosd(angle),sind(angle)];
        end
    if
sum(polyxpoly([q_old(1),q_new(1)],[q_old(2),q_new(2)],x_obs,y_obs))~=0
% if collision with obstacle break out of while loop
         break
    end
    Tree_init=[Tree_init;q_new]; % add q_new to tree
```

```
    Edge_init=[Edge_init;q_old,q_new]; % add branch to edge
   plot([q_new(1),q_old(1)],[q_new(2),q_old(2)],'k');
    end
   %% Connect q_goal
    [IDX,D_goal]=knnsearch(Tree_init(:,1:2),q_goal);
    if min(D_goal)<20 % threshold value to try to attempt goal to
branch, follows same architecture as above
        q_near=Tree_init(IDX,1:2);
        angle=atan2d([q_goal(2)-q_near(2)],[q_goal(1)-q_near(1)]);
        q_new=q_near;
        while sum(q_new~=q_goal)~=0
            q_old=q_new;
            d=pdist([q_new;q_goal]);
            if step_size>d
                q_new=q_goal;
            else
                q_new=q_new+step_size*[cosd(angle),sind(angle)];
            end
            if
sum(polyxpoly([q_old(1),q_new(1)],[q_old(2),q_new(2)],x_obs,y_obs))~=0
                break
            end
            Tree_init=[Tree_init;q_new];
            Edge_init=[Edge_init;q_old,q_new];
            plot([q_new(1),q_old(1)],[q_new(2),q_old(2)],'k');
        end
    end
    if q_new==q_goal % break out of for loop if goal is connected
        break
    end

end

%% path
path=[];
q=q_goal;
while sum(q~=q_init)~=0 % work backwards through branches to go from
goal to initial location

[A,B]=ismember(q,Edge_init(:,3:4),'rows');
parent=Edge_init(B,1:2);
path=[path;q];
q=parent;
end
path=[path;q];
plot(path(:,1),path(:,2),'r')
```

RRT* MATLAB script

```
% rrt_star.m
% conducts RRT star algorithm to find optimal path
clc; close all; clear;

% vehicle start location
```

```
x_init=10;
y_init=1;

step_init=1; % step size
step_size=step_init;
n=500; % number of attempts to expand the tree
pathplot=[];
min_path=1000;
edges=[];
path=[];
Radius=40;
figure()
hold on

% workspace
xworkspace=[0;0;20;20;0];
yworkspace=[0;20;20;0;0];
plot(xworkspace,yworkspace)

% obstacle
x_obs=[0,0,1,1,0,0,2,2,0]*2+9;
y_obs=[0,1,1,5,5,6,6,0,0]*2+5;
fill(x_obs,y_obs,'b')
%%

% initial and goal

q_init=[y_init,x_init];
plot(q_init(1),q_init(2),'ro')
q_goal=[18,10];
plot(q_goal(1),q_goal(2),'ro')
% Build RRT
Tree_init=[q_init];
Edge_init=[q_init,q_init,0];
Tree_goal=[q_goal];
Edge_goal=[];
for i=1:n


    collision=1;
    % Select a random collision free point
    while collision~=0
    q_rand=[randi(20*n/10)/(n/10),randi(20*n/10)/(n/10)];
    collision=sum(inpolygon(q_rand(:,1),q_rand(:,2),x_obs,y_obs));
    end

    % Extend RRT
    [IDX,D]=rangesearch(Edge_init(:,3:4),q_rand,Radius);
    IDX=cell2mat(IDX);
    D=cell2mat(D);
    q_near=Edge_init(IDX(1),3:4);
    angle=atan2d([q_rand(2)-q_near(2)],[q_rand(1)-q_near(1)]);
    q_new=q_near;
    h=[];
```

```
    counter=1;
    step_size=step_init;
    while sum(q_new~=q_rand)~=0

        q_old=q_new;
        [A,B]=ismember(q_new,Edge_init(:,3:4),'rows');
        d=pdist([q_new;q_rand]);
        if step_size>d
            q_new=q_rand;
            step_size=d;
        else
            q_new=q_new+step_size*[cosd(angle),sind(angle)];
        end
        if
sum(polyxpoly([q_old(1),q_new(1)],[q_old(2),q_new(2)],x_obs,y_obs))~=0
            break
        else
            Tree_init=[Tree_init;q_new];
            Edge_init=[Edge_init;q_old,q_new,step_size+Edge_init(B,5)];
            counter=counter+1;
        end
    end


%% rewiring of tree
[IDX,D]=rangesearch(Edge_init(:,3:4),q_new,Radius);
IDX=cell2mat(IDX);
D=cell2mat(D);
min_cost=Edge_init(end,5); % current cost is connected edge, search to
find a shorter one
change=0;
    for j=1:numel(IDX)
        cost=D(j)+Edge_init(IDX(j),5);
        if cost<min_cost % search for shorter collision free path
            if
sum(polyxpoly([q_new(1),Edge_init(IDX(j),3)],[q_new(2),Edge_init(IDX(j)
,4)],x_obs,y_obs))==0
                change=1;
                row=IDX(j);
                min_cost=cost;
            end
        end

    end
    if change==1
    Edge_init=[Edge_init;Edge_init(row,3:4),q_new,min_cost];
        if counter>1 % delete old branches
            Edge_init((end-(counter-1)):end-1,:)=[];
            counter=1;
        end
    end

end
 %% q_goal star
```

```
[IDX,D]=rangesearch(Edge_init(:,3:4),q_goal,Radius);
IDX=cell2mat(IDX);
D=cell2mat(D);
change=0;
for j=1:numel(IDX) % searches through edges to find minimum cost path
    cost=D(j)+Edge_init(IDX(j),5);
    if cost<min_path
        if
sum(polyxpoly([q_goal(1),Edge_init(IDX(j),3)],[q_goal(2),Edge_init(IDX(
j),4)],x_obs,y_obs))==0
            change=1;
            row=IDX(j);
            min_path=cost;
        end
    end
end
if change==1
    if Edge_init(row,3:4)~=q_goal
        Edge_init=[Edge_init;Edge_init(row,3:4),q_goal,cost];
    end
    path=[];
    q=q_goal;

    while sum(q~=q_init)~=0 % searches through parents to make path
        [A,B]=ismember(q,Edge_init(:,3:4),'rows');
        parent=Edge_init(B,1:2);
        d=pdist([parent;q]);
        path=[path;q,d];
        q=parent;
    end
    d=pdist([parent;q]);
    path=[path;q,d];
    pathplot=plot(path(:,1),path(:,2),'g','LineWidth',2);
    counter=0;
    change=0;

end

% waypoint paths
x_path=flipud(path(:,2));
y_path=flipud(path(:,1));
```

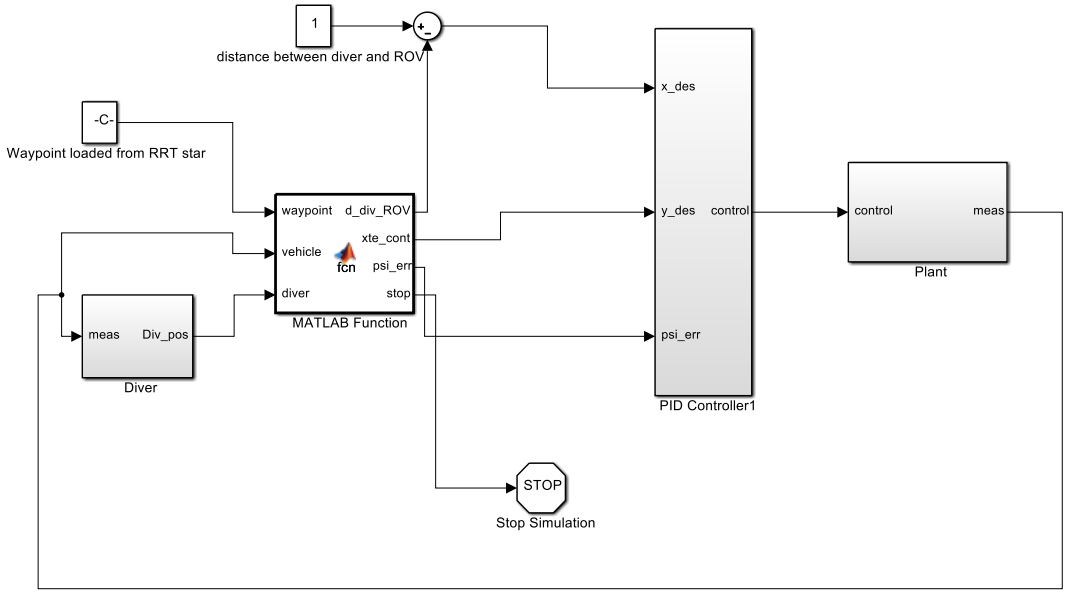

Figure 51. Simulink diagram of RRT algorithm

MATLAB script for waypoint following

```
function [d_div_ROV,xte_cont,psi_err,stop] =
fcn(waypoint,vehicle,diver)
%#codegen
x=vehicle(1);
y=vehicle(2);
psi=vehicle(3);
persistent i change
rho=.1; % watch circle radius (m)
if isempty(i)
    i=2;
    change=0;
end

num_waypoints=size(waypoint); % determine number of waypoints
num_waypoints=num_waypoints(1);



distance_sq=(waypoint(i,1)-x)^2+(waypoint(i,2)-y)^2; %squared distance
psi_track=-atan2(waypoint(i,1)-waypoint(i-1,1),(waypoint(i,2)-
waypoint(i-1,2)))+pi/2; % track angle
Li=sqrt((waypoint(i,1)-waypoint(i-1,1))^2+(waypoint(i,2)-waypoint(i-
1,2))^2); % distance between waypoints
psi_CTE=psi-psi_track+beta; % cross track error angle
xtilde=(waypoint(i,1)-x); % difference between current position and
next waypoint in x
ytilde=(waypoint(i,2)-y); % difference between current position and
next waypoint in y
S=(xtilde*(waypoint(i,1)-waypoint(i-1,1))+ytilde*waypoint(i,2)-
waypoint(i-1,2))/Li^2; % track line
```

```
del_p=atan2(waypoint(i,1)-waypoint(i-1,1),waypoint(i,2)-waypoint(i-
1,2))-atan2(xtilde,ytilde); % angle between line of sight and track
line
xte=S*Li*sin(del_p);
xte_cont=xte;% cross track error
psi_des=-atan2((waypoint(i,1)-x),(waypoint(i,2)-y))+pi/2; % desired
heading
% determines side of track vehicle is on
if xte>0
LOS=psi_track+pi/2;
else
    LOS=psi_track-pi/2;
end
if abs(xte)>1||change==1 % performs LOS control
psi_des=LOS;
xte_cont=0;
change=1;
end
if abs(xte)<0.5&&change==1
change=0;
xte_cont=xte;
end
% angle goes from -pi to +pi
if psi_des>pi
    psi_des=psi_des-2*pi;
elseif psi_des<-pi
    psi_des=psi_des+2*pi;
end
  psi_err=psi-psi_des;
if psi_err>pi
    psi_err=psi_err-2*pi;
elseif psi_err<-pi
    psi_err=psi_err+2*pi;
end

if distance_sq<=rho^2
    i=i+1; % next waypoint
end
d_div_ROV=sqrt((diver(1)-vehicle(1))^2+(diver(2)-vehicle(2))^2);

% goal reached
if i==num_waypoints+1
stop=1;
i=i-1;
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] C. Georgiades et al., "AQUA: An aquatic walking robot," in *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems,* Sendai, Japan, 2004, pp. 3525-3531.

[2] G. Dudek et. al., "A visually guided swimming robot," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems* 2005, pp. 3604–3609.

[3] *Water –running robots*. [Online]. Available: http://www.rutgersprep.org/kendall/7thgrade/cycleC_2009_10/dr/Aquarobots.html

[4] J. Vaganay et al., "Ship hull inspection with the HAUV: U.S. Navy and NATO demonstrations results," in *OCEANS 2006,* 2006, pp. 1–6.

[5] D. J. Stilwell and B. E. Bishop, "Platoons of underwater vehicles," *IEEE Control Systems,* vol. 20, no. 6, pp. 4552, Dec. 2000.

[6] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on Robotics and Automation,* vol. 14, no. 6, pp. 926–939, Dec. 1998.

[7] M. A. Lewis and K. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous Robots,* vol. 4, no. 4, pp. 387-403, Oct. 1997.

[8] H. M. Choset, *Principles of Robot Motion: Theory, Algorithms, and Implementation.* Cambridge, MA: MIT Press, 2005.

[9] L. Barnes, M. Fields and K. Valavanis, "Unmanned ground vehicle swarm formation control using potential fields," in *Mediterranean Conference on Control & Automation, 2007,* Athens, Greece, 2007, pp. 1–8.

[10] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research,* vol. 5, no. 1, pp. 90-98, 1986.

[11] F. E. Schneider and D. Wildermuth, "A potential field based approach to multi robot formation navigation," in *Proceedings. 2003 IEEE International Conference on Robotics, Intelligent Systems and Signal Processing,* Changsha, China, 2003, pp. 680-685.

[12] M. Porfiri, D. G. Roberson and D. J. Stilwell, "Tracking and formation control of multiple autonomous agents: A two-level consensus approach," *Automatica,* vol. 43, no. 8, pp. 1318–1328, 2007.

[13] S. M. LaValle, *Planning Algorithms.* New York, NY: Cambridge University Press, 2006.

[14] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research,* vol. 17, no. 7, pp. 760-772, 1998.

[15] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine,* vol. 4, no. 1, pp. 23-33, 1997.

[16] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research,* vol. 30, no. 7 pp. 846-894, 2011.

[17] *vLBV Operators Manual*. D ed., SeaBotix., San Diego, CA, 2013.

[18] *SeaBotix*. [Online]. Available: http://www.seabotix.com

[19] *Vicon Engineering.* [Online]. Available: http://www.vicon.com/Application/Engineering

[20] J. Weiss and N. Du Toit, "Real-time dynamic model learning and adaptation for underwater vehicles," unpublished.

[21] N. S. Nise, *Control Systems Engineering.* Hoboken, NJ: John Wiley & Sons, 2004.

[22] T. I. Fossen, *Guidance and control of ocean vehicles.* New York, NY: John Wiley and Sons, 1994.

[23] D. Horner, "Path Following and Autopilots," unpublished.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California